NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

MIDS UNIVERSAL INTERFACE FOR THE SELECTION OF LINK 16 MESSAGES

by

Milton E. Mata

March 2002

Thesis Co-Advisor: Valdis Berzins
Thesis Co-Advisor : Luqi

Approved for public release; distribution is unlimited.

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	March 2002	Master's Thesis

4. TITLE AND SUBTITLE
MIDS Universal Interface for the selection of Link 16 messages

5. FUNDING NUMBERS

6. AUTHOR(S)
Milton E. Mata

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Naval Postgraduate School
Monterey, CA 93943-5000

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES
The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release, distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT

Department of Defense is required to provide MIDS terminals for subsurface, surface and airborne weapon systems. The MIDS terminals use legacy software that does not have the flexibility to support multiple platform interfaces. With evolving interface requirements, there is a need to accommodate more platforms that require a new interface design software approach.

This thesis proposes a new software interface design approach to effectively support the emergent need of various platforms to integrate with the MIDS terminal. The current designs of the MIDS interfaces are not flexible and are very costly to support new emergent requirements. This thesis provides a framework, which consist of a Graphical User Interface (GUI) that allows the developer to create new platform interface specification. Once the interface has been created, the automation of code generation can be achieved.

14. SUBJECT TERMS
Control Panel, Embedded Systems, Framework, Source Code, Interface, Universal, Interoperability, Software, Graphical User Interface

15. NUMBER OF PAGES
157

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL

NSN 7540-01-280-5500
Standard Form 298 (Rev. 289)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited.

MIDS UNIVERSAL INTERFACE for the SELECTION of LINK 16 MESSAGES

Milton E. Mata
B.S.E.E., Florida Atlantic University, 1981

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN SOFTWARE ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
March 2002

Author: Milton E. Mata

Approved by: Valdis Berzins, Thesis Co-Advisor

Luqi, Thesis Co-Advisor

Luqi, Chair
Software Engineering Curriculum

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

Department of Defense is required to provide MIDS terminals for subsurface, surface and airborne weapon systems. The MIDS terminals use legacy software that does not have the flexibility to support multiple platform interfaces. With evolving interface requirements, there is a need to accommodate more platforms that require a new interface design software approach.

This thesis proposes a new software interface design approach to effectively support the emergent need of various platforms to integrate with the MIDS terminal. The current designs of the MIDS interfaces are not flexible and are very costly to support new emergent requirements. This thesis provides a framework, which consist of a Graphical User Interface (GUI) that allows the developer to create new platform interface specification. Once the interface has been created, the automation of code generation can be achieved.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF FIGURES

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEGEMENT

I would like to thank Dr. Valdis Berzins, Dr. Nabendu Chaki and Dr. Jun Ge for their guidance, support and invaluable help for making this thesis possible. I would like to acknowledge my Program Manager Captain John Kohut and the late Mr. Kim Laporte for their assistance.

Special thanks to colleagues and friends that made this thesis possible: Steve Goff. Finally to my parents, my wife Christina and my son Ernesto Jose thanks for your support and words of encouragement.

I. INTRODUCTION

A. MULTIFUNCTIONAL INFORMATION DISTRIBUTION SYSTEM (MIDS) DESCRIPTION

The MIDS PMW-101 is the International Program Office that is in charge of the development of the MIDS terminals. The MIDS is a terminal that operates in a Link-16 network, Figure 1.1 illustrates a typical Link-16 network; the MIDS terminal allows the processing of Link 16 messages and is widely used by many nations and services for their weapon systems. For this thesis, when I refer to weapon system, platform or host I refer to systems that use/integrate the MIDS terminals for the processing of Link-16 messages.

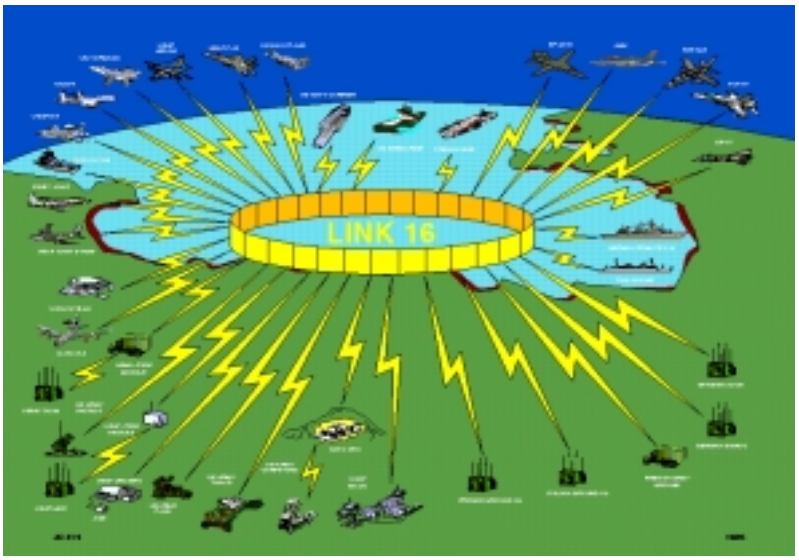

Figure 1.1 Typical Link-16 Network.

The MIDS consists of five Computer Software Configuration Items (CSCIs): the Core CSCI, Tailored Input and Output (TIO) CSCI, the Voice CSCI, Message (MSG) CSCI and TACAN CSCI. The MIDS terminal has four basic interfaces but can use only

one interface at the time, in other words when the 1553 bus [ref 1] is active the other interfaces are off and do not perform any message processing.

The Multifunctional Information Distribution System (MIDS) is used for Link 16 communication. The Link 16 includes the TADIL-J Message catalog. The following messages are part of this catalog: Network Management; Precise Participant Location and Identification; surveillance; Antisubmarine warfare; intelligence; information management; weapons coordination and management; control; platform and system status; electronic warfare; national use and miscellaneous messages. All of these message types composed the superset of messages that I have defined as the universal message set (refer to page 39, section V), the message set satisfies all platform requirements.

Figure 1.2 illustrates the MIDS architecture and the interface to the platform or host. The microprocessors in the MIDS are connected via the VME bus. The software interface between these CSCIs is as follows:

The Core CSCI generates and processes messages for transmission for the MSG CSCI.

The MSG CSCI performs the encryption and decryption of messagess for the Core CSCI processing.

The Voice CSCI sends data to the Core CSCI; the Core CSCI processes these messages and provides the required response.

The TIO CSCI sends messages to the Core CSCI; the Core CSCI processes these messages and sends the appropiate input to the TIO CSCI.

The TACAN interfaces with the TIO CSCI and provides the necessary flight control data for the Core and TIO CSCIs.

The Core CSCI and TIO CSCI reside in two different MC68040 microprocessors, a portion of the TIO CSCI resides in the avionics mux (MC6883 microprocessor) and ground mux resides in the MC6836 microprocessor. The Voice CSCI resides on TMS32C50 microprocessor, the message CSCI resides in the Intel 80386 microprocessor and the TACAN CSCI resides in TMS320C31.

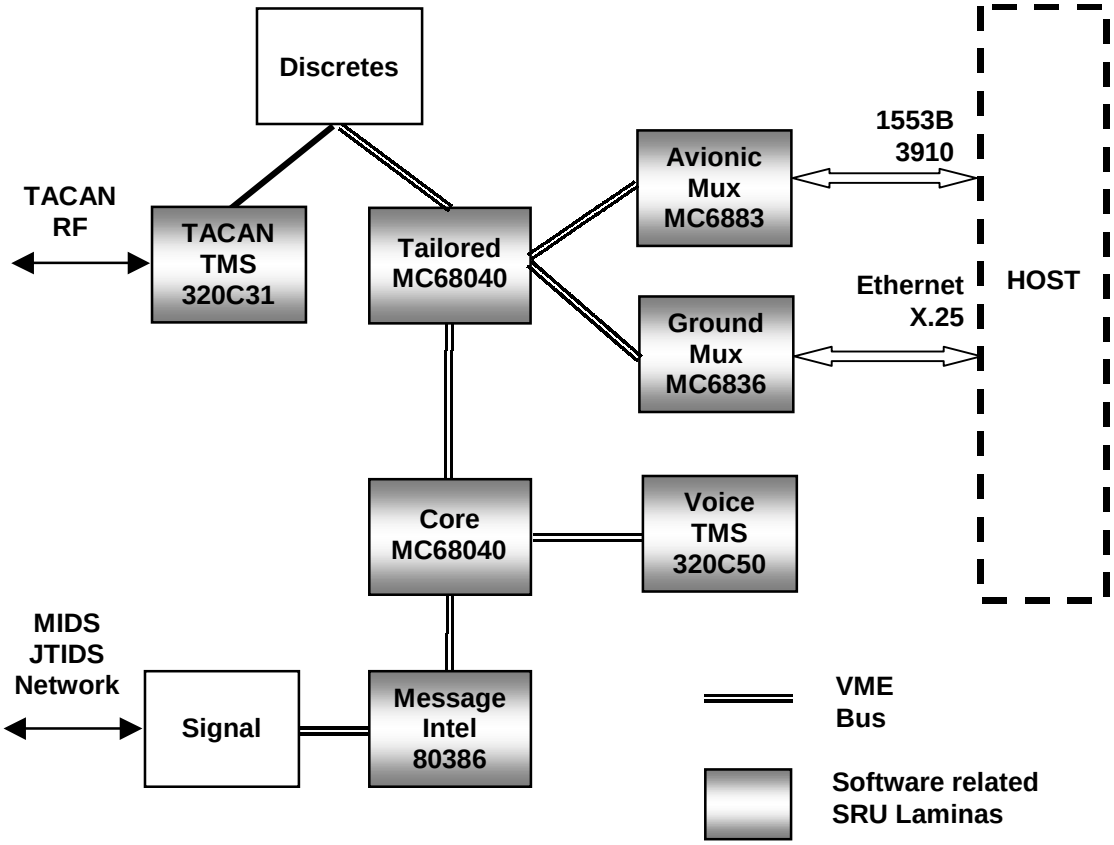

Figure 1.2 MIDS Architecture and Interface to Host.

Figure 1.3 illustrates the message flow between the Core CSCI and the TIO CSCI and the TIO CSCI and the host or weapon system. Note that for simplicity, the Voice, Message and TACAN CSCIs are not shown in this figure.

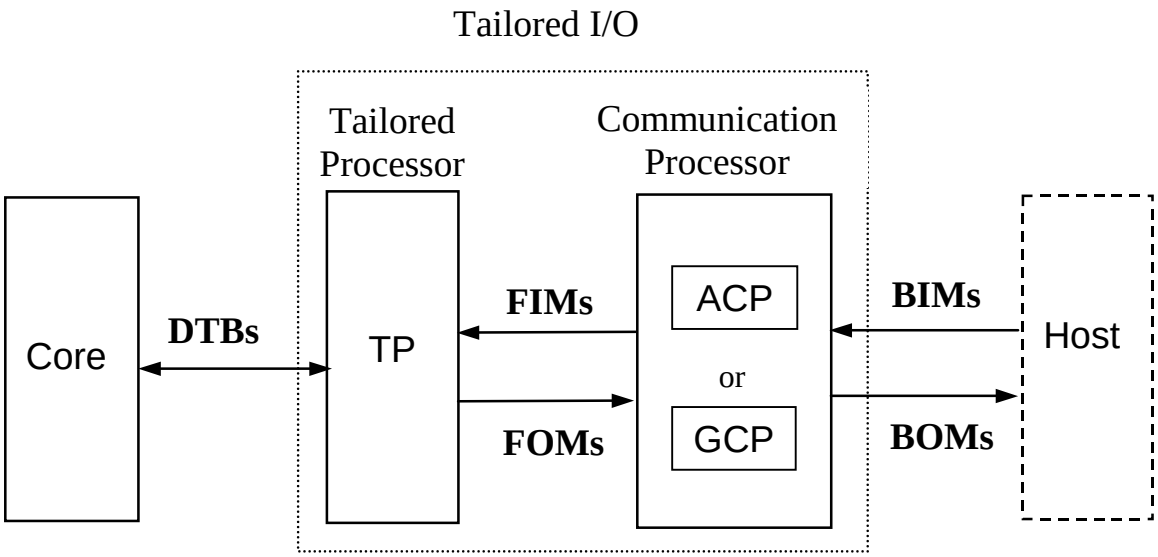

Figure 1.3 MIDS Interface and Message Flow.

Figure 1.3 shows the transmission and reception of different type of messages:

The messages between the Core CSCI and Tailored Processor are called Data Transfer Blocks (DTBs). The DTBs can be of variable length and represent the information received from the host. The Core CSCI processes this information and sends information back to the Tailored I/O via the DTB.

The messages between the Tailored Processor and Communication processor are called Functional Input Messages (FIMs) and Functional Output Messages (FOMs) [ref. 2]. These messages are independent of the physical media and are composed of 16 bits words; the messages can be of variable length. Each FIM or FOM can include information such as navigation; TACAN control mode or other MIDS related type of data.

BIMs are the input messages coming from the host and BOMs are characterizes as the output messages from the MIDS LVT to the host, the BIMs and BOMs are part of the the physical layer. The FIMs and FOMs are mapped into the BIMs and BOMs [ref. 2]

The Core and the TIO are the most important CSCI of the MIDS terminal. These MIDS CSCIs allow the basic Link 16 communication processing and allow the weapon systems to be able to participate in a Link 16 network.

The TIO originally was designed to interface with four different buses (1553, 3910, Ethernet and X25), the TIO provides the communication link between the MIDS terminal and the host platform, the TIO passes Link 16 messages to the weapon systems [ref. 2].

For each existent interface, we have special TIO software associated with the interface, in other words there is unique code and set of messages associated with the type of interface. A unique code is associated with the 1553 interface, the X25, 3910 and Ethernet, but many of these messages that these platforms use are common to the different interfaces [ref. 2].

In addition to these four basic interfaces, the MIDS terminal includes different types of interfaces for the 1553 platforms; the message for each platform varies for the different types of 1553 interfaces (Figure 3.1 shows the existent 1553 bus interfaces). The TIO software processes these messages and provides these inputs to the host platform. The TIO CSCI gets the information from the platform and processes the messages received from the weapon system, the TIO CSCI sends Link-16 messages to the host computer via their interface.

The US Department of Defense has designated Link-16 as the primary tactical data link for Service, and Defense Agency Command and Control and Intelligence and has mandated the use of these system where is practical. The MIDS terminal is a software intensive system and is the backbone of Link-16. In order to meet the requirements of the weapon systems, periodic software upgrades are required. There is a need to use software-engineering disciplines to properly support the MIDS life cycle.

B. PROBLEM STATEMENT

Initially, the MIDS terminal interfaces were designed for four different platforms; each interface had a unique message set associated with it. However, when new platforms were added and needed to use the MIDS terminal interface and its message set, it was discovered that we did not have all the correct messages that were needed for the new platform's integration and use. For that reason the integration between the MIDS and the platform types could not be implemented. Thus, a new platform type and a new set of messages needed to be developed to meet the platform requirements.

In order to accommodate the new platforms, additional TIO software and TIO test software were needed to properly integrate the MIDS with the new platform types. In many instances, only one or two messages had to be added to the existent message set for the proper integration. Since the TIO software was not flexible enough to accommodate these changes, a new platform type needed development. The program office had to task a contractor to develop the new interface software, new test software and associated documentation.

The development of this unique software was acceptable for two or three platforms. But since the MIDS terminal is integrated with many other platforms, this approach is impractical and expensive. Currently, there is a requirement to integrate various weapon systems with the MIDS terminals while the weapon system interface should be customized to fit for the MIDS terminals. The customization of this interface costs time and money and delays the proper modernization of existent weapon systems. As an example, The F-16 wanted to upgrade their avionics system with the MIDS terminal. When F-16 was developing the interface between the F-16 avionics system and the MIDS terminal, they discovered that they did not have a platform type associated with them and that a number of Link -16 messages required to perform their mission were not available in the existing interfaces. Due to this deficiency, the integration between the F-

16 (or platform “I”) [ref. 1] avionics system and MIDS terminal had to be delayed until a new interface was developed and tested.

The objective of this thesis is to provide a reusable abstract interface that can be adapted to various system requirements. The reusability of the existing interface component provides potential savings in the development effort of new complex interfaces.

C. 1553 BUS PROTOCOL

The 1553 is a dual redundant bus architecture that provides aircraft with internal division command/response multiplex capability. The 1553 is composed of two birectional lines (for redundancy). Each of those lines is direct coupled, twisted shielded pair. The transmission rate is 1 Mbit/s). The 1553 bus is used to connect the electronic equipment in an aircraft [ref. 1]. Typically, sensors, radios (such as the MIDS) and dual redundant mission computers are connected to the bus; the mission computer controls the bus and handles all the data received from these devices.

In my thesis, I call the platforms that use the 1553 bus, 1553 platforms. The F/A-18, F-16, AMX Tornado and EF-2000 are identified in my thesis as 1553 platforms. F/A-18 is called platform “A”, EF-2000 is called platform “B”, AMX Tornado is called platform “F”, F-16 is called platform “I”, and the EF-2000 is called platform “L” [ref. 1].

D. TIO SOFTWARE CURRENT DEVELOMENT STRATEGY

The current software development strategy allows the duplication of source code for 1553 or Ethernet interfaces even if minor changes are made to the software. As an example, the majority of the platform “A” [ref. 1] interface source code was duplicated in order to meet the integration requirements of platform “I”, minor modifications were done to this software in order to meet the unique characteristics of platform “I”. Therefore, there is duplication of software and testing when a new platform is added. Table 1.1 illustrates the commonality of messages between platforms “A” and “I” [ref. 1].

For simplicity, I am showing the Functional Interface Messages (FIMs) used by platform "A" and "I", several of these FIMs are common but other FIMs are not. FIMs are defined in on page 4, section I.

FIM ID	Platform "A"	Platform "I"
FIM01	Used	Used
FIM02	Used	Used
FIM03	Used	Used
FIM04	Used	Used
FIM05	Used	Not used
FIM06	Used	Not used
FIM07	Used	Not used
FIM08	Used	Not used
FIM09	Used	Not used
FIM12	Used	Not used
FIM17	Used	Not used
FIM18	Used	Not used
FIM21	Not used	Used
FIM25	Used	Not used
FIM26	Used	Not used

Table 1.1 FIMs Used by Platform A and B.

After an analysis of both interfaces, it was discovered that the platform "A" source code needed minor modifications in order to meet the platform "I" requirements. Since the TIO architecture could not be easily modified without the potential risk of introducing significant errors into the program it was decided that the software needed to be

duplicated and added to the baseline. The contractor had to make minor changes to the duplicated software, retest the software, and delivery it to the program office.

At the present time, we add a new interface to the MIDS terminal every time a platform is added, with the new interface there is changes associated to the TIO CSCI, the test software and associated documents.

This acquisition strategy is costly and has a schedule impact on the MIDS terminal integration with weapon systems. As an example, it took six months to develop the TIO software for platform "I" at a cost of $1 million dollars. The cost and extra time affects the readiness of the armed forces. In addition, the Central Processor Unit (CPU) uses more time to execute the program, memory usage is increased and risk is added to the software development cycle due to the new requirements. The current CPU loading is approximately 65% of its capacity, the maximum accepted loading capacity for the CPU is 70%, and the MIDS terminal is rapidly approaching this threshold. The numbers of platforms are constantly increasing and the addition of new interfaces, increase the time loading factor of the CPU thus affecting the performance of the MIDS terminal.

E. PURPOSE

The MIDS TIO software has deficiencies that make the integration between the MIDS and the platform expensive and time consuming. I propose changes to the TIO software framework in order to correct these deficiencies and make the TIO software less rigid and versatile. This thesis proposes the development of software that can allow the user to configure interfaces easily. The framework allows more flexibility to the MIDS interface.

Existing software techniques such as generative programming [ref 3], feature models [ref. 3] and rapid prototyping [ref. 4] can be used to develop a more flexible MIDS interface. Since the MIDS is a family of products, the MIDS can use this technique to model platform types and the messages associated with the platform. In addition, the development of a prototype allows me to model this interface and to

demonstrate that my approach is feasible and that the interface can be made more flexible. The current technology to achieve this flexibility is available. For this thesis, I have explored the current technologies and possible application to the MIDS interface; my goal is to develop an approach that can use these techniques to develop an interface that can be used by many platforms.

F. EXPECTED SOLUTION AND MOTIVATION

Since the MIDS terminal is widely by the US and NATO, it is worthwhile to investigate and propose a framework that eliminates the need for a new interface and new test software when a new platform is added. In addition to the new framework, I would like to use generative programming [ref. 3] to obtain certain degree of automation for the platform type

In order to improve the current situation, I have designed a framework that allows me to reconfigure the interface of different types of platforms without having to change the TIO software or modifying the test software. Code can be generated automatically [ref. 3] to meet this requirement. The new framework can be tested after it has been finilized and after that the software is tested only if a new message is used or the architecture is changed.

The new framework facilitates the integration between the MIDS terminal and platform. My work allows the program office to save time and money and can eliminate the need to maintain additional software interfaces. My approach can improve the current software situation by eliminating the need of additional software development.

If this universal interface is developed for the current MIDS hardware, there can be significant savings to the US and NATO when new platforms are integrated with the MIDS terminals. The schedule to integrate the MIDS terminal and the host computer could decrease and software can be reused.

A solution to this problem is to define a more flexible interface that meets the emergent requirements of a weapon system. The basic requirement is to develop a

universal interface capable of selecting platform types, messages types and generating Link-16 messages.

In my approach, I identified a universal set of messages required (page 37, section V) for this interface. The need for developing a prototype to demonstrate this concept is useful to show the feasibility of my approach and the processing of Link-16 messages.

My software prototype [ref. 4] approach allows the user to design an interface from an interface panel. The interface panel lets the user to choose a platform and set of messages from the MIDS Interface Control Panel. After the selection is processed a new platform type would have been created, thus making the interface very flexible. My prototype addresses the weapon systems that use the 1553 interface [ref. 1], since it is the interface that is used by the majority of the weapon systems and until the present time is the interface that the MIDS terminals are using the most.

The program offices, the integrator, and the users need the universal interface in order to perform their platform integration to save money and time. Software engineering techniques bind the design approach and allow us to use existent interface frameworks. Design techniques such as feature models can be used to model the MIDS interface, since the MIDS terminals are a product of families. These techniques and methodologies can be used to modify existent frameworks and interfaces to meet our current requirements.

G. UNIVERSAL INTERFACE DESCRIPTION AND EXTENSIBILITY

The defined universal interface has all possible messages used to interface the MIDS to other platforms. This interface has been tested to make sure that the messages are processed correctly. I have used different message combinations and message sets to validate the various interfaces.

Since the platforms do not use all the messages in the message superset, it is adequate to test one or two message subsets to validate the software. I identified this subset of messages from the superset of messages (page 37, section V) to prove that the

new platforms can use these messages and be able to accomplish their mission. The test cases for this message include the subset of messages that I want to validate.

The interface can be tested when the initial framework is developed but after that there is no need to test the software any further, since a combination of messages can be used to prove that the TIO software works properly.

The proposed model can be extended to all message sets required by the MIDS and the users. The US Army or US Air Force can use these configurations without waiting for the contractor to modify and test the software.

H. FOCUS

I focused on the definition of a new software interface to meet the emergent needs of software platforms. This thesis includes a new framework that allows a more flexible interface.

The new framework includes the processing of Link-16 messages for the 1553 interface [ref. 2]. By creating this superset of messages, a platform can have the capability of choosing the messages that they want from this superset of messages, without having to develop new software for the TIO CSCI and its associated test software.

The 1553 interface [ref. 2] currently uses the subset of the messages identified on page 37, section V. This platform does not have a need for the entire set of messages. The messages used by the different platform interfaces are identified in the MIDS Interface Control Document [ref. 2].

As part of my research, I am planning to investigate and use as applicable, current technology in areas such as software wrappers [ref. 2], rapid prototyping [ref. 2], frameworks [ref 5], generative programming [ref. 3], interoperability [ref 4], and requirement definition [ref. 4]. I have used this information in my thesis as a basis for the proposed approach and recommendation.

I. KEY TECHNICAL CHALLENGES

I have identified the following key technical challenges in my thesis; the technical challenges are as follows:

1. Definition of Requirements

A complete understanding of the platform message requirements is a must. This ensures that a full set of messages for the different weapon systems is defined.

2. Framework Definition

The framework has to meet the emergent requirements of new platforms. I have identified the risks associated with the redefinition of the new framework. The interface can be as flexible as possible to allow for the creation of a new interface.

3. Rapid Prototyping

By using rapid prototyping, I have to prove that the new framework is feasible and it can be used for the creation of platforms and associated messages.

4. Testing

I have developed two test cases to test the proposed framework; the test cases can verify the performance of the existent Link-16 messages. The test cases can verify the two different platform configurations.

The messages that I have selected under the following categories: Network Management; Precise Participant Location and Identification; and platform and system status category. I have identified the specific messages in Figure 5.1 page xx of this thesis.

J. OBJECTIVES OF SOLUTIONS

The objective of the solution is for other platforms to be able to choose or select a complete set or subset of existent messages; these messages can be tailored in accordance with the platform's requirement. The user selects from a table the list of message(s) that

they want to use in order to meet their operational requirements without having to wait for a period of time before the software is developed and tested.

When the MIDS Universal Interface Control Panel is developed, the user chooses the new platform types without having to worry about development of new software or development of new test cases for the software or a set of different configurations.

The MIDS Universal Interface Control Panel (refer to page 27, in section III for a pictorial) software was tested once to demonstrate the proper performance and after that no further tests are planned. This is due to the fact that the different set of messages would have been fully tested after the MIDS Universal Interface framework has been changed and tested by the developer.

K. STRUCTURE OF THESIS

This thesis is organized as follows:

Chapter II addresses the current methods and technologies available and a summary of the current technology; and the applicability of this technology to MIDS interface.

Chapter III addresses the TIO Feature model, Generative programming, Interface problems, objectives of proposed solutions and framework.

Chapter IV addresses the implementation, framework concerns and shortcomings, rapid prototyping and development of test cases.

Chapter V addresses the unclassified message set.

Chapter VI addresses the interface case description to the platform under test and the test limitation.

Chapter VII addresses the conclusions, the future work and the summary of my conclusions and findings.

II. CURRENT TECHNOLOGY

A. SURVEY OF CURRENT METHODS/TECHNOLOGIES

In order to support my thesis, I have looked into the current technology trends and I have identified several techniques that I can use in order to support my development effort. I have researched several textbooks, articles and journals on Generative programming [ref. 3], wrappers [ref. 4], rapid prototyping [ref. 4], frameworks [ref. 5], and testing [ref. 6]. These articles have given me an understanding of the status of the current technology and allowed me to use part of their findings to formulate my approach in resolving the MIDS interface problem. In the following sections, I addressed the status of this technology and the manner in which I can apply this technology to the MIDS Universal Interface.

B. REVIEW OF CURRENT TECHNOLOGY

1. Architecture and Frameworks

The software architecture is the set of all the concepts (i.e., software components, frameworks, database, paradigm, programming language and other) used to "view" the entire software before it is designed [ref. 6].

The software architecture of a program or computing system is the structure or structures of the system, which comprise software components, the Externally visible properties of those components, and the relationships among them [ref. 6].

"Externally visible" properties, refers to those assumptions that other components can make of a component, such as its provided services, performance characteristics, fault handling, and shared resource usage. The intent of this definition is that a software architecture must abstract away some information from the system and yet provide

enough information to be a basis for analysis, decision making, and hence risk reduction [ref. 6].

The architecture defines components and embodies information about how the components interact with each other. This means that architecture specifically omits content information about components that does not pertain to their interaction [ref. 6].

Systems can comprise more than one structure, and no one structure holds the irrefutable claim to being “the architecture”. By intention, the definition does not specify what architectural components and relationships are. Is a software component an object? A process? A library? A database? A commercial product? It can be any of these things and more [ref. 6].

Every software system includes an architecture, and every system can be shown to be composed of components and relations among them [ref. 6].

The behavior of each component is part of the architecture, insofar as that behavior can be observed or discerned from the point of view of another component. This behavior is what allows components to interact with each other, which is clearly part of the architecture. Hence, most of the box-and-line drawings that are passed off as architectures are in fact not architectures at all. They are simply box-and-line drawings.

Framework is a reusable design expressed as a set of abstract classes and the way in which their instances collaborate with each other.

Framework models are similar to the structural view, but their primary emphasis is on the coherent structure of the whole system, as opposed to concentrating on its composition. Framework models often target specific domains or problem classes. Work that exemplifies the framework view includes domain-specific software architectures [ref. 6].

The advantages of using frameworks are that they are more flexible than they have been in the past. Current frameworks allow us to put a system together in a more coherent manner, information can be hidden within classes and software components can be used by different software packages without having to retest the software components.

The messages that are passed between classes make the maintenance of software much easier and less susceptible to errors.

One of the disadvantages of a new framework or modification to the framework is that testing is required to verify that the components are properly integrated. It takes time and resources to develop the components of a new framework. The development of a new framework adds risks to a program, especially to those programs that are using legacy software.

2. Generative Programming

This is a technique that allows the user/host to create a set of specifications for their platform type. The generative programming captures the configuration knowledge in a program form and allows the users to create a family member rather than having to look for a particular component in the source code [ref. 3].

The idea of generative programming is to build generative models for families of systems and generate concrete systems from these models. Implementation and configuration knowledge can be reused for each generated member [ref. 3].

One of the advantages of generative programming is that it uses software building blocks to facilitate the development of software. The objects have an identity and behavior associated with them and since these blocks are based on specification performance it makes the software more modular and easier to maintain. In addition, generative programming reduces the amount of new code that needs to developed and tested. In other words, human errors are reduced while the operational and test software is being developed [ref. 3].

I have used templates to drive the code generation; this is the key for generating the output. The template provides the necessary controls for choosing the specification or the types of services that the users want to define as the new interface. The templates can be extended and created easily.

One of the disadvantages is that the developing of a new architecture or framework is expensive. There is no reliable data that can be used in order to determine

where the break point is or if it is worthwhile to use an approach that has not been proven yet.

a) Features and Features Models

Feature and Feature Models are used in domain analysis to identify the differences and common features of a system. These concepts were introduced by the Featured Oriented Domain Analysis method [ref. 3].

The features of software systems are identified as follows:

(1) Feature diagrams is used to represent if the feature is mandatory or if it is optional [ref. 3].

(2) Feature definition identifies the features that are compiled at different stages of execution [ref. 3].

(3) Composition rules identify the features that are valid [ref. 3].

(4) Feature rational provides the rational for the features that are chosen [ref. 3].

One of the advantages of features and features models is that allow us to reuse common software modules and to build from these software commonalties new features, thus providing significant savings to the development of software. Unique characteristics of the software can be modeled and easily understood.

One of the disadvantages is that when using legacy software, the legacy software cannot be characterized easily. Legacy software of older systems cannot be easily modeled in this manner. Due to the lack or incorrect software documentation, legacy software cannot be easily understood. The software redesign is very expensive and the return in investment may not be seen immediately but after a few years.

3. Rapid Prototyping

This technique is used to explore the range of possible solutions, normally in this environment a subset of functions or requirements can be implemented in a less rigorous

environment, more resources are available and risks associated with the development effort are diminished [ref. 4].

While software prototyping, the engineers can evaluate and measure the performance of a system. At the same time, the user can evaluate and decide whether or not this is the proper approach and is able to recommend a possible solution. In the case of the MIDS Universal Interface, the user may be able to choose the messages from an interface control panel. Thus I have gained the ability to determine if the code generator meets the specified requirements [ref. 4].

Rapid prototyping allows me to make decisions on concrete data rather than trying to imagine a system that does no exist and since software is very abstract it is very difficult to convince potential users to follow this path [ref. 4].

One of the advantageous of rapid prototyping is that allow us to determine the feasibility of the software development approach, the constraints associated with the software and we gain a better understanding of the software prototype [ref. 4].

One of the disadvantageous of software prototyping is that the prototype may not represent the full system, or the prototype does not run on the desired platform, or run at the correct speed [ref. 4].

4. Software Testing

The software is tested on the software building blocks. Test cases for each software component is developed in order to test the software at the component level, and then a test case are developed when the software is integrated [ref. 6]. Several platform types are tested to verify that the messages include the proper information (the messages that are chosen by the user or host). The testing is model based; I will be looking for inputs and states that will trigger error conditions in the proposed model and prototype.

One of the advantages of software testing is that I can establish a degree of confidence on the software that I have developed. I have been able to verify that the representation of the requirements meets the expected result.

One of the disadvantages is that the framework cannot be fully tested and I am not able to run the software on the hardware. The MIDS Interface Control Panel software runs on a Personal Computer.

5. Wrappers

Software wrapping is a technique that is created around existing software; it provides a new view to external systems, objects or users. Wrapping can be done at multiple levels. The intent of the wrappers is to access the functionality of legacy software while increasing functionality and user base. Wrapper can create objects that store a primitive value and contain methods for working with the primitive value [ref. 4].

C. SUMMARY OF CURRENT TECHNOLOGY SURVEY

The technology survey provides me with the necessary techniques to design the framework, compile the interface according to the specifications, and design the test cases to test the new framework. The generative programming allows me to compile the software according to my specifications, and rapid prototyping allows me to prove that these techniques work for the MIDS universal interface.

I believe that generative programming, rapid prototyping and other technogy can be used to improve the current TIO design and make it more flexible. This technology offers a more viable approach to maintain legacy software, since it introduces techniques that allow the software developer to model a framework that can be changed easily. By designing a code generator, framework and a prototype, I can prove that the code generator can be used to write the source code for new platforms that require different Link-16 messages.

However, the new techniques introduce risks to real time system such as the MIDS. As an example, wrappers cannot be used due to its overheads, the MIDS current design uses up to 80 % of the CPU time. Software redesign also includes changing the architecture, re-testing software and often building prototypes to demonstrate the concept,

this implies that funding is required for the new development and with funding cuts, money may not be available for software development.

D. APPLICABILITY TO THE MIDS UNIVERSAL INTERFACE

I have used the different techniques and methods included in section B of this chapter.

- Current frameworks and templates will be used to develop a new TIO framework to develop the code for the auto auto generated code.
- Generative programming will allow me to specify the requirements that I want to implement in the interface
- Rapid prototyping will allow me to prove that the concept works and can be modeled
- Testing will allow me to test the code that will be generated by the code generator for at least one new platform.

A modified framework will be used to meet the emergent requirements of new platforms. The modified framework will have capability to support the existent platform types too. The superset of messages could be used by any platform.

Other techniques, such as wrappers were considered but not used for my prototype. I can not use wrappers for the Link-16 messages due to the high overhead and the impact on the performance of TIO CSCI. For time critical system, the data may become stale in messages and this information does not represent the current status (as an example, a navigational solution may old and can provide an incorrect position for platforms).

This thesis provides a solution to the current problems and allowing us to proceed with an approach that will save time and money when new platforms integrate the MIDS terminals into their weapon systems.

THIS PAGE INTENTIONALLY LEFT BLANK

III. MIDS FEATURE FRAMEWORK AND MESSAGE DEFINITION

A. TIO FEATURE MODEL

The Feature and Feature Models technique will be used to model the MIDS Universal Interface according to its feature and service requests, the different interfaces can be modeled as features, such as the 1553 interface or the interface chosen by the user. The feature model will allow the user the flexibility that is required to support future platforms.

MIDS Universal Interface can be represented as a feature model since MIDS Universal Interface has distinctive interface features (Figure 3.1 illustrates this model). This figure shows the existent interfaces (1553, 3910, X25 and Ethernet interfaces). Furthermore, figure 3.2 denotes the existent 1553 interfaces and future 1553 interfaces (this is denoted as Platform "new" in figure 3.2) will be derived from this interface.

In my thesis, I am interested in all future 1553 interfaces, for simplicity platform "A" will correspond to platform 00 in my MIDS Universal Control Panel. 1553 bus interfaces can use up to 32 bus sub-addresses; platforms could use the same sub-address field and its associated Link-16 messages. Unfortunately, the Link 16 message requirements for the platforms using the 1553 bus interface change with time. When the changes on the Link 16 message requirement take place, platforms cannot longer use the same set of Link-16 messages and software modifications are needed. It is important to remember that only one feature or interface can be active at the time. The universal set of Link-16 messages is defined in chapter V of this thesis, this message set will be used to compose a platform, and the messages can be manually selected via the interface control panel.

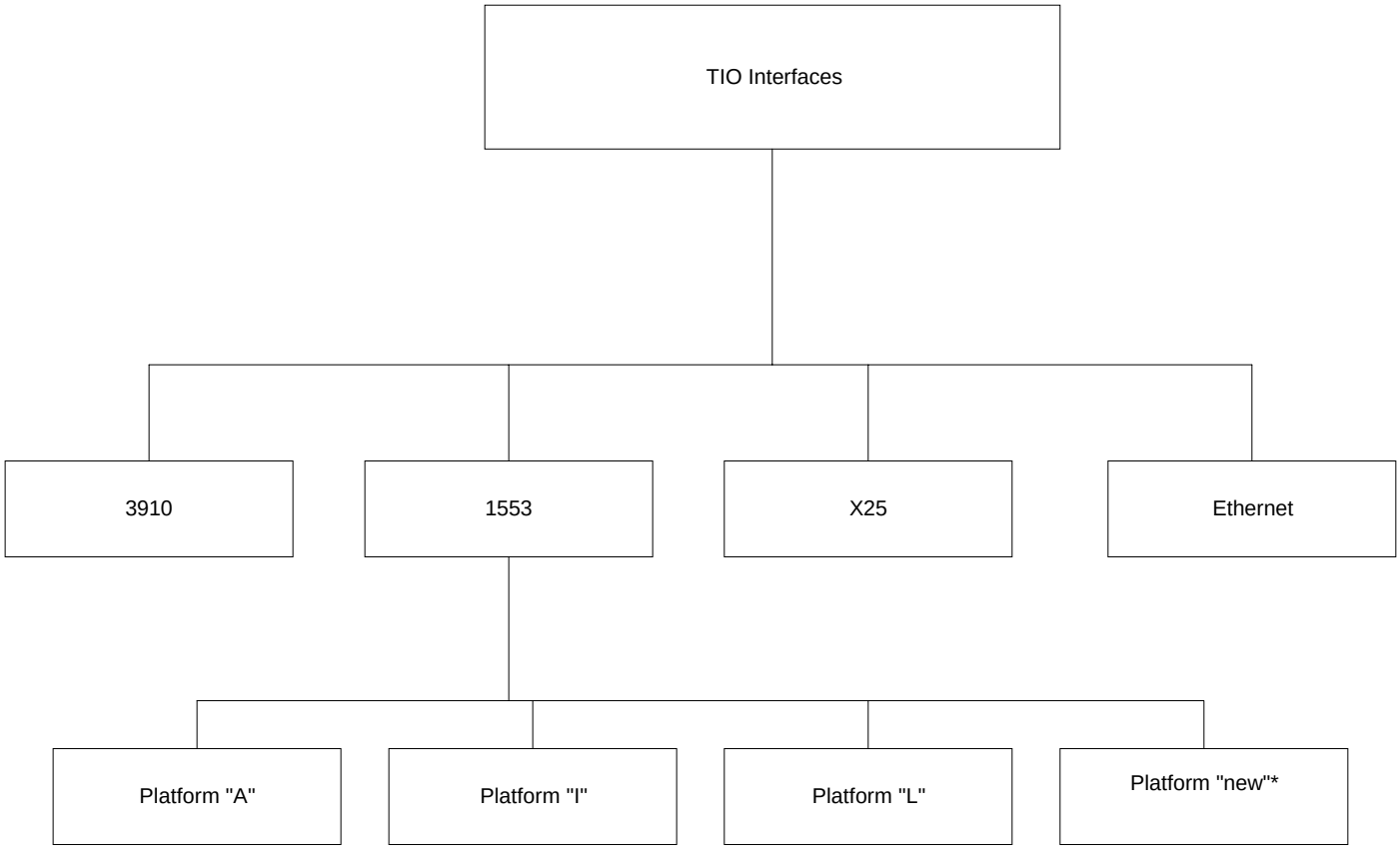

Figure 3.1 MIDS Current Interfaces and Future Interfaces.

Figure 3.1 shows the different types of interfaces, including the different 1553 interfaces. Table 3.1 shows the current MIDS interfaces and the platforms that use these interfaces.

Interfaces	Platform Types
1553	A, B, I, L
3910	C
X 25	D
Ethernet	E

Table 3.1. TIO Current Interfaces.

B. TIO GENERATIVE PROGRAMMING

The generative programming technique will be used to auto generate code for each platform type and the message contents. As an example, if I chose messages FIM01 through FIM20 in section V, Table 5.1, I will be able to generate the message contents when I press the code generator key in the MID Interface Control Panel. Chapter V defines the MIDS Interface Control Panel and its output, the messages can be selected manually and using this approach creates a new platform type. Once the platform type and associated messages are created, a new file will be created that will show the new platform type and its messages. The new interface will be tested to demonstrate the program will support the development of new platforms.

Once the user selects these messages, the selected messages represent a platform type that has a unique use for these messages. As an example, a platform type can use messages 1 through 20, this new platform may represent platform "10" or if another platform uses messages 2 through 31, this platform may represent platform "15"

After the software is compiled, the software will be tested to verify that the correct messages have been compiled (according to the specified configuration) and the proper message information is passed through these messages. The system components and interfaces will be tested to validate the defined configuration.

C. INTERFACE DEFINITION GOALS

The MIDS interfaces do not work with all the 1553 platforms that want to integrate the MIDS with their platform. Each platform requires a unique interface or set of messages that are not currently provided by the MIDS. The MIDS provides a predefined set of interfaces that do not meet the 1553 interface requirements of future weapon systems.

Therefore, I have developed a framework (refer to page 30 in sction 3) that allows the user to select their messages of interest. Figure 3.2 illustrates a potential interface; this interface includes the shaded messages FIM01 and FIM03 that may be needed for a platform ID 25, the platform that I want to create for this example. By using the MIDS Interface Universal Control Pane, I can create an interface with an ID of 25 and messages FIM01 and FIM03. However, I can create a platform 25 with messages FIM02 and FIM10. At the present time, we do not have this flexibility, and this is the flexibility that I intend to achieve with this framework, the framework allows the user to configure any platfrorm.

FIM ID	Message Name	Message Selected
FIM01	Common Carrier FIM	x
FIM02	Init & Status Data Request FIM	x
FIM03	Init Data Change FIM	x
FIM04	External Time Reference – Mode 2 FIM	
….	….	….
….	….	….
FIM34	Short R/C response	x

Table 3.2 Message Selections

Table 3.2 illustrates the different choice of messages that are available to me. The Message Name column describes the name of the FIM; the "x" in the Message Selected column is used for illustration purposes and denotes that the message(s) has been selected. For my prototype design, I have chosen not to display the Message Name and Message Selected column.

D. LINK 16 UNIVERSAL MESSAGE DEFINITION

I have created a set of messages (page 37, section V) from the MIDS Interface Control Document [ref. 1] and I have composed the list of messages from the different interfaces and combined all of these messages into a Link-16 Universal Message Set (refer to page 37, section V for this definition). This message set is the baseline for all 1553 platform types that I have created for this thesis, the users is allowed to choose from this set of messages and configure their platforms accordingly.

E. FRAMEWORK

The framework is designed to resolve the interface problems associated when a new platform type is created. My framework advocates the use of auto code generation; the generated software can be compiled according to the messages specified by the user in their configuration, after the software is compiled then the platform will be able to use the new software configuration. Note that only one platform can be configured at the time and only one message set will be available to work with that particular interface.

I have used a personal computer to develop the software for the MIDS Interface Control Panel. The prototype software runs on Windows 98 or Windows NT, the prototype software is coded in Java and a hexadecimal editor and a text editor to check the results of the test.

The framework allows the user to select a platform via a GUI. The GUI allows the users to create a new platform type, associate the messages with the platform and auto generates the code.

The GUI is needed to facilitate the configuration of the new platforms and for generating the necessary code for the platform. This is a big improvement over the current design where new software needs to be developed for weapon systems that want to use the Link-16 capability. By using my approach, the software can be developed one time and there is not need for further software development efforts. The current software

development approach does not meet the weapon system needs and my approach resolves these problems.

1. MIDS Universal Control Interface Panel

The MIDS Universal Control Interface Panel is a window that gets displayed when the Mids_interface_rev7.java file is executed (refer to figure 3.2, page 28 for the display of the MIDS Interface Control Panel). The window includes two comboboxes, a button that generates the code, information text boxes. This window has the properties of a standard window. Refer to appendix D, page 137 for more details.

The MIDS Universal Control Interface Panel introduces three new software functions that makes the 1553 interface more flexible, these functions are as follows: “Select platform type”, “Select Messages” and “Code generator”. These functions are available to the user by pressing a button on the interface panel. These three new functions allow us to change, modify and create new platforms. MIDS Universal Control Panel is illustrated in figure 3.2 on page xx; this figure shows the main components of the panel and the current available data entries. The following paragraphs describe the function of each key and the action(s) required by the user.

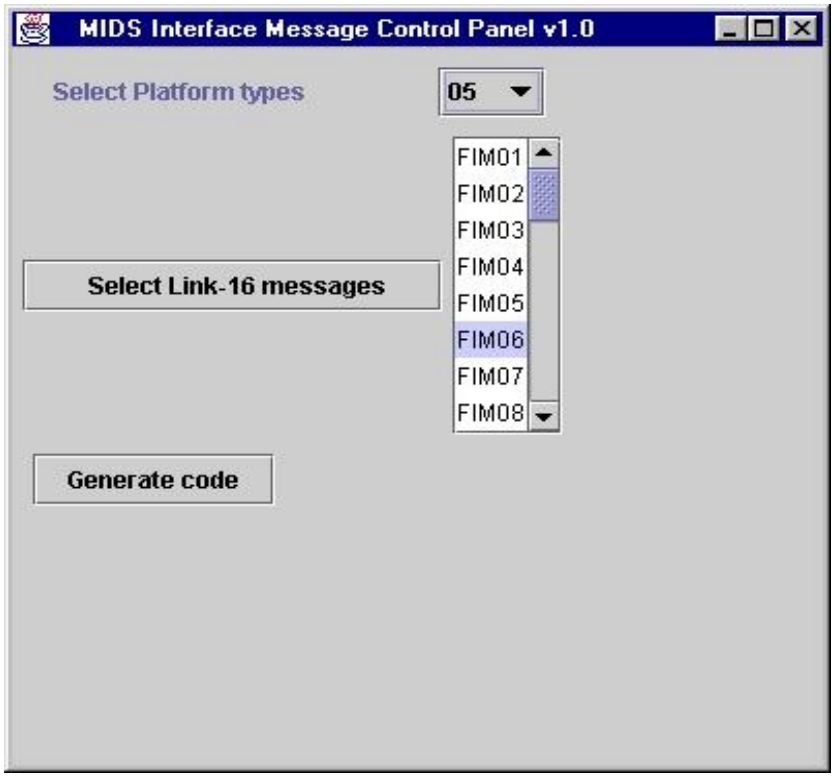

Figure 3.2 MIDS Universal Control Interface Panel.

a) The Select platform type key

The Select platform type key allows the processing of the platform number. The user can select the platform number from 00 to 31; this is the valid range for 1553 interfaces. The default platform number is 05, the platform type can be selected by clicking the mouse on the platform number, and the user can scroll up and down on the combo box. The user can select the platform by clicking on the platform number (refer to figure 3.2).

b) The Select message key

The “Select message key” allow the user to choose messages from a list of messages, these messages have been named FIM(s), and the user can select one or many FIMs from this list. There are a total of 34 FIMs as shown in the Figure 5.1 page xx, the user can scroll up and down the window to select the messages, and the default message is FIM06. The selected messages are stored in a queue until the user presses the code generator key. Multiple messages can be selected by holding the CTRL key on the keyboard and by clicking the mouse.

c) The Code generator key

The Code generator takes the platform number and its chosen messages and generate a file with the contents of the messages. The platform and message contents are written in a file to demonstrate the processing of the platform and its set of messages. As soon as the messages are selected, the messages are put in a cue and stay there until the Code generator key is activated.

2. MIDS interface Control Panel Description

Figure 3.3 shows the use case for the MIDS Control Panel, it shows the user and the basic operations performed by the user.

The user is allowed to select, create platform, generate code, and manage the platform and messages. The user gets feedback on the DOS window and the files that are created when the key for the code generator has been pressed.

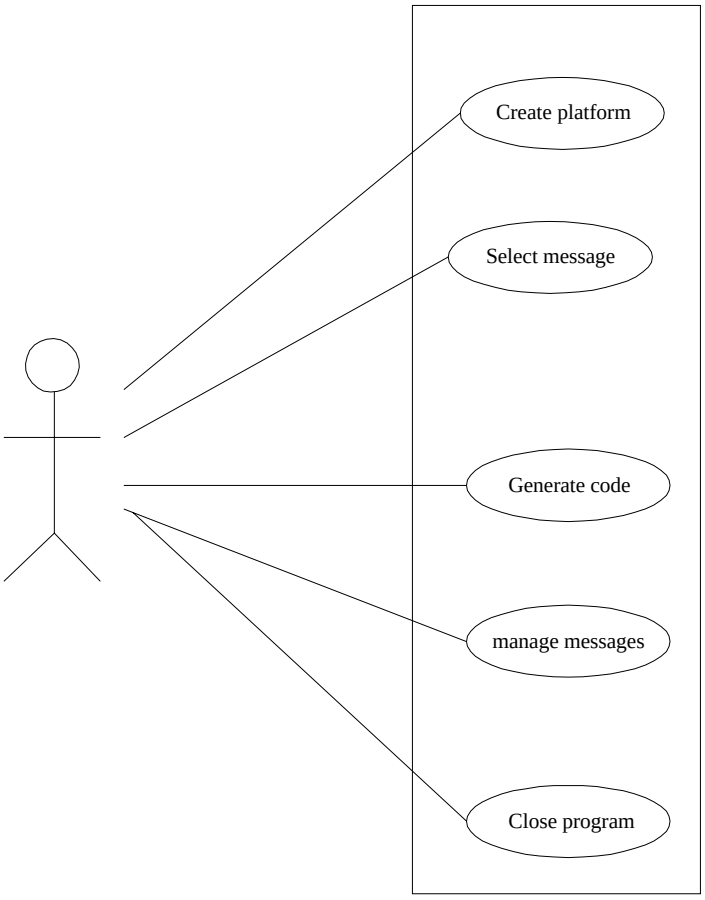

Figure 3.3 MIDS Interface Control Panel Use case.

3. MIDS Interface Generator Framework

The framework that processes the creation of platform types, the framework consists of four distinctive components; the main components are the GUI, the platform number, the associated messages and the code generator. The compiler is shown as part of the framework but is not physically connected to the GUI.

Figure 3.4 shows the main components of the framework and the relationship between the components.

The universal message set look up table is part of the framework, message look up table includes the 34 FIMS. The user selects messages from the list, when the user

finishes compiling the desired list then he can go to next step. The next step is to choose the platform number.

The selected platform allows the user to select the platform number, when the selection is completed the user is ready to generate the code for the message contents.

After the compiler has compiled the software then the software is tested to verify that the interfaces work properly and to gain confidence on the quality of the software. Initially the software will be released to the platforms for their use. The compiled software has been tested in two different configuration or platform types, after the initial tests are performed; the software developer does not have to test software for future platforms.

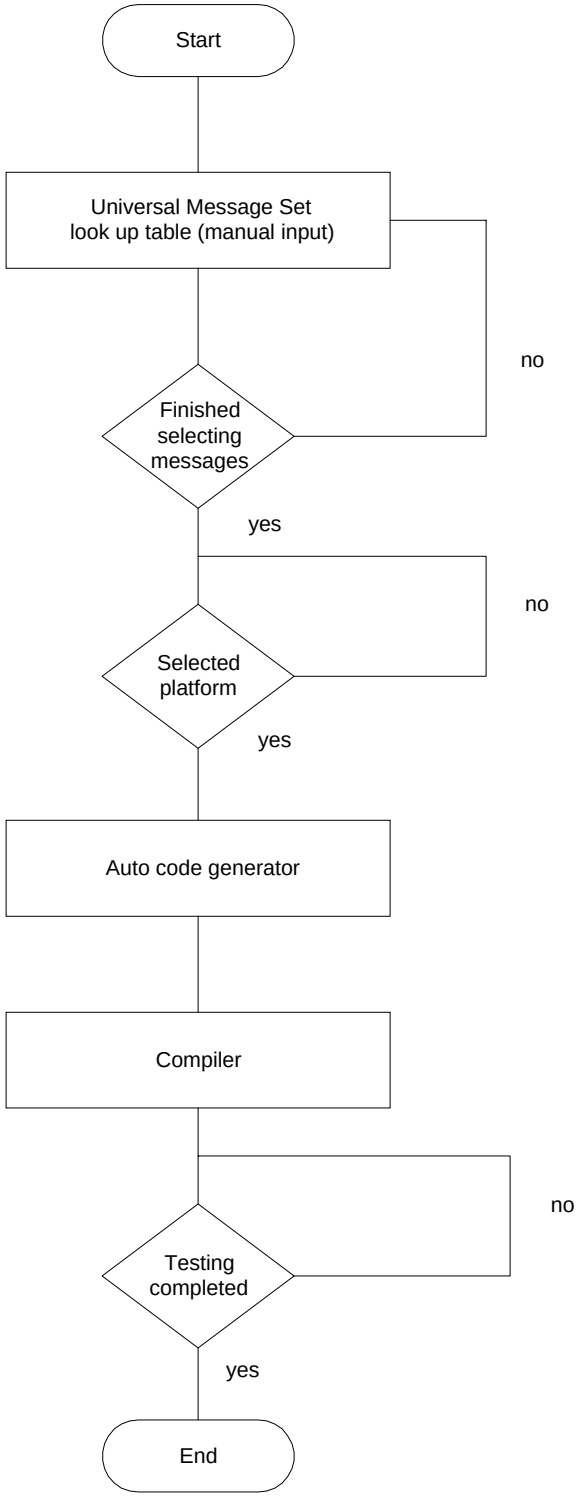

Figure 3.4 MIDS interface Control Panel Framework.

THIS PAGE INTENTIONALLY LEFT BLANK

IV. IMPLEMENTATION

A. IMPLEMENTATION USING THE PROPOSED FRAMEWORK

I have developed a GUI interface (refer to Appendix "A" for the source code) to compose the different types of 1553 interfaces. For this thesis, I have written the software (refer to appendix A, page for the listing) for the MIDS Interface Control Panel and associated functions (select platform, select messages and generate code). The GUI is window based and allows the user to select the proper parameters.

The MIDS Universal Interface Control Panel includes two classes; these classes handle the different events generated by the MIDS Universal Control Panel. The classes are Mids_interface_rev7 and class Fim. Figure 4.1 on page xx shows the illustration of these two classes:

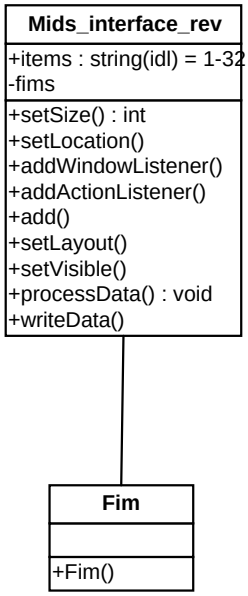

Figure 4.1 MIDS Interface Class Diagram

The GUI software includes the necessary methods to draw the GUI and perform platform ID and message management. The following functions perform window management functions:

The Mids_interface_rev7 class includes the following methods:

- setSize(), this method sets the size of the window.
- setLocation(), this method sets the location of window on the display.
- addwindowListener(), this methods listens for window events such as close or resize the window.
- addActionListener(), this method listens for mouse events (clicking the mouse) and detects when the Code generator is pressed.
- add(), this method is in charge of displaying the data entry menus. The argument for the add() function changes (i.e., displaying the FIMs number or platform number).
- setLayout(), this method sets the component locations on the panel.
- SetVisible(true), this method sets the display to true.
- ProcessData(Object platform, Object [] fims), this method keeps track of the selected platform number. In addition, this method saves the values of the selected fims and process and keeps the contents of each fim. Initializes the the fimCodes and saves the selected values for the fim codes after initialization,
- writeData(fimcode), this method get the fimCodes values.
- WriteData(Fim []fims), this method creates a new file with the message contents and checks if errors in the file manipulation have occurred.
- File("FimTemplate.java), this method creates a java file for the different FIM classes (from FIM01 through FIM 34), each class includes a set of methods that builds word 0, word 1, and word 2.
- Random.Access.File (), this is an overloaded method that uses several argument to classes, methods, variables and the necessasy data for the creation of a template. This template handles all the messages created by ther user. This function creates the following classes, methods and variables:

- Java packages for proper compilation
- Class FimTemplate, this is the name of the file that is being create by the code generator
- GetWord0(), this is the method that gets word 0 of the different types of FIMs
- GetWord1(), this is the method that gets word 1 of the different types of FIMs
- GetWord2(), this is the method that gets word 2 of the different types of FIMs
- BuildWord0(), this method builds the contents of word 0 for each FIM class
- BuildWord1(),this method builds the contents of word 1 for each FIM class
- BuildWord2(),this method builds the contents of word 2 for each FIM class
- GetName(), this method gets the name of the FIM,
- WriteData, this method writes the FimClasses
- SetName(), this method sets the name of FIM message

The Fim class includes the following methods:

- Fim(), this method initializes the Fim's name, word0, word1, word2 and word3.

B. FRAMEWORK CONCERNS

The TIO CSCI software runs on dedicated hardware and is hardware dependent, the TIO CSCI has many time constraints that have not been included in this framework. I chose to model only the interfaces to the Avionics and Ground devices.

I cannot demonstrate the performance of this framework on the real system, since the hardware is not available to me. I do not have the system to monitor or check the contents of these messages. However, the bit information can be checked in a file that the interface control panel can create.

I chose to model the FIMs (refer to page 4 for a definition of FIM, section I), for the TIO CSCI, my framework does not model the FOM.

I can have 2^{32} potential interfaces, these interfaces can be associated with a different type of message thus creating different types of configurations that can not be fully tested, especially when each words is composed of many bits. It is impossible to create these many interfaces and create test scenarios to exercise the software

C. FRAMEWORK SHORTCOMINGS

The framework addresses the creation of interfaces according to the selected messages but the MIDS Interface Control Panel does not process any of these messages once they have been transmitted or received.

The framework should be able to handle the interface to other CSCIs. The interfaces to the other CSCIs are not shown in this framework for simplicity.

In addition, the Program Office may be reluctant to change to a new framework design without assessing the potential cost and risks associated with a new development.

D. RAPID PROTOTYPING FEASIBILITY

The MIDS Universal Interface can be prototyped using existent software packages and software techniques. I used Java to write my software and I was able to download the software from the Internet for free.

The GUI and Link 16 messages are easy to implement. Auto code generators are very popular and are available. I have no plans to use any new languages or technology to develop the software for the MIDS Universal Interface. However, several Link 16 messages are very long, require a lot of coding time and I cannot emulate every single bit value in the message.

The risk associated for the development of this software prototype is medium and requires a working knowledge of Java.

V. MIDS UNIVERSAL INTERFACE DEFINITION

A. UNCLASSIFIED MIDS LINK-16 MESSAGE SET

Table 5.1 includes the messages that I am using for the different interfaces and for the code generation. Each message represents the information that is generated for the MIDS terminal interface and it is sent to the host computer.

The messages are called FIMS, the FIMs represent data structures, made up of 16 Bit words for data to be exchanged with the host at the functional level. FIMs are aggregates of data elements, gathered by functional cohesion, that satisfy the exchange requirements. The definition of FIMs also contains the layout of these data elements and does not depend on the medium used to convey them.

In order to define the Link 16 universal message set, I used the MIDS Interface Control Description to compile a list of all the messages used by the different interfaces. Each platform may use between 12 to 15 messages, not all the platforms use the same FIMs, as an example platform “00” may use FIMs 01 through 04, FIM10, FIM12, FIM27 and FIM28, the remaining FIMs may not used by this platform. Figure 5.1 identifies this list of messages and the name of the message. These messages are of variable length and are composed of data words, each word is 16 bits long. MIDS information is defined in each word, such as message ID or R/C message ID, this information and bit decomposition is included in the MIDS Interface Control Document. For my prototype, I decided to use the first three words of each message, since the messages can be very long.

A data element is a piece of information that cannot be split for the MIDS. A unique identifier, a bit-level representation and a functional description define a data element.

	FIM ID	FIM Name
1	FIM01	Common Carrier FIM

2	FIM02	Init & Status Data Request FIM
3	FIM03	Init Data Change FIM
4	FIM04	External Time Reference – Mode 2 FIM
5	FIM05	Request For Test Message FIM
6	FIM06	Request For RTT-A Range FIM
7	FIM07	MIDS Delay Reduction FIM
8	FIM08	Request For Route Establishment FIM
9	FIM09	Variable Message Format (VMF) FIM
10	FIM10	Host Navigation Data Type 1 FIM
11	FIM11	External Time Reference – Mode 1 FIM
12	FIM12	TACAN Control Type 1 FIM
13	FIM13	TACAN Control Type 2 FIM
14	FIM14	Request for PPLI Transmission FIM
15	FIM15	Track Data Base Amplification Data Request FIM
16	FIM16	Host Navigation Data Type 2 FIM
17	FIM17	Host Navigation Data Type 3 FIM
18	FIM18	Host Navigation Data Type 4 FIM
19	FIM19	Host Navigation Data Type 5 FIM
20	FIM20	TACAN Control Type 3 FIM
21	FIM21	Maintenance Parameters Change FIM
22	FIM22	TACAN Control Type 4 FIM
23	FIM23	Host Navigation Data Type 6 FIM
24	FIM24	Host Navigation Data Type 7 FIM
25	FIM25	Host Navigation Data Type 8 FIM
26	FIM26	Host Navigation Data Type 9 FIM
27	FIM27	Short Initialization Data Change 1 FIM
28	FIM28	Short Initialization Data Change 2 FIM
29	FIM29	Host Navigation Data Type 10 FIM
30	FIM30	Host Navigation Data Type 11 FIM
31	FIM31	Host Navigation Data Type 12 FIM
32	FIM32	Host Navigation Data Type 13 FIM
33	FIM33	Host Navigation Data Type 14 FIM
34	FIM34	Short R/C Response

Table 5.1 Link 16 Universal Message

B. MESSAGE DESCRIPTION

The following paragraphs describes the FIM functions:

1. Common Carrier FIM

This FIM shall be used to convey to the Terminal Free Text messages, Link 16 Fixed Format messages, or IJMS Fixed Format messages for transmission over the MIDS network.

This FIM is also used to convey control actions to the Terminal, in which case the contents of the message are not being transmitted over the network and the transmission control fields are useless.

2. FIM02: Initialization & Status Data Request FIM

This FIM shall be used by the Host to request the content of either: the Terminal current initialization data, the Terminal maintenance parameters, the Terminal status data, the Terminal initialization data sets, and the Terminal Core processor memory contents.

3. FIM03: Initialization Data Change FIM

This FIM shall be used to change Terminal initialization data., initialization data may be changed in two ways: via initialization loads, where the Terminal accepts a set of complete or partial initialization data blocks before performing validity checking and implementing new values. Initialization data loads allow the Host to load up to 8 initialization data sets; via initialization data changes, where the Terminal accepts, validity checks; and implements changes on specified words of the current initialization data.

The Initialization Data Change FIM, according to the different values of the LOAD COMMAND field, allows the Host: to initiate and control initialization data load processing, to perform initialization data changes and Initialization data changes are ignored by the Terminal during the start-up phase.

4. FIM04: External Time Reference - Mode 2 FIM

This FIM is used by the Host to convey the external time reference mode 2 TOD to the Terminal.

5. FIM05: Request For Test Message FIM

This FIM is used by the Host as a stimulus for the Terminal to generate a Communication Control message requesting the transmission of a Test Message.

6. FIM06: Request For RTT-A Range FIM

This FIM is used by the Host as a stimulus for the Terminal to generate and transmit a Link-16 type 2(0) RTT interrogation message in the RTT-A Range PG.

7. FIM07: MIDS Delay Reduction FIM

This FIM shall be used to convey to the Terminal Link 16 Fixed Format messages to be transmitted over the MIDS network IAW MIDS Delay Reduction processing.

8. FIM08: Request For Route Establishment FIM

This FIM is used by the Host as a stimulus for the Terminal to generate and transmit a route establishment message with the Action Index set to "Route Request". The message is transmitted on the ROUTE ESTABLISHMENT PG, using classical re-promulgation.

9. FIM09: Variable Message Format (VMF) FIM

This FIM shall be used to convey to the Terminal all VMF messages received from the Host for transmission over the MIDS network.

10. FIM10: Host Navigation Data Type 1 FIM

This FIM is used by the Host to convey Host Navigation Data to the Terminal.

11. FIM11: External Time Reference - Mode 1 FIM

This FIM is used by the Host to convey the external time reference mode 1 TOD to the Terminal.

12. FIM12: TACAN Control Type 1 FIM

This FIM shall be used to convey TACAN control data.

13. FIM13: TACAN Control Type 2 FIM

This FIM shall be used to convey TACAN control data. The content of this FIM shall overwrite the content of the corresponding fields in the TACAN Control Word 1 of the Initialization Data File.

14. FIM14: Request For PPLI Transmission FIM

The host uses this FIM as a stimulus for the Terminal to generate and transmit a PPLI message.

15. FIM15: Track Data Base Amplification Data Request FIM

This FIM shall be used to convey an amplification data request for a defined track database entity. The Terminal responds to this FIM with Track Data Base Amplification Data FOM.

16. FIM16: Host Navigation Data Type 2 FIM

This FIM shall be used to convey the Host Navigation Data for a Laser Inertial Navigation System (LINS) to the Terminal.

17. FIM17: Host Navigation Data Type 3 FIM

This FIM shall be used to convey the Host Navigation Data from the Flight Control Computer (FCC) to the Terminal.

18. FIM18: Host Navigation Data Type 4 FIM

This FIM shall be used to convey the Host Navigation Data (GPS) to the Terminal.

19. FIM19: Host Navigation Data Type 5 FIM

This FIM shall be used to convey the Host Navigation Data (FIX) to the Terminal.

20. FIM20: TACAN Control Type 3 FIM

This FIM shall be used to convey TACAN control data.

21. FIM21: Maintenance Parameters Change FIM

This FIM shall be used by the Host to reprogram the value of the Terminal Maintenance Parameters. Maintenance Parameters can be reprogrammed with this FIM either via the support port interface or via the data bus interface with the host.

22. FIM22: TACAN Control Type 4 FIM

This FIM shall be used to convey TACAN control data.

23. FIM23: Host Navigation Data Type 6 FIM

This FIM shall be used to convey the Host Navigation Data (GPS) to the Terminal

24. FIM24: Host Navigation Data Type 7 FIM

This FIM is used by the Host to convey Host Navigation Position Data to the Terminal.

25. FIM25: Host Navigation Data Type 8 FIM

This FIM is used by the Host to convey Host Navigation Velocities Data to the Terminal.

26. FIM26: Host Navigation Data Type 9 FIM

This FIM shall be used to convey the Host Navigation Data (FIX) to the Terminal.

27. FIM27: Short Initialization Data Change 1 FIM

This FIM is used to convey to the Terminal up to 15 segments (contiguous 16-bit data words) of initialization data. The position of each segment in the Init Data File is defined in Maintenance Parameters Data Words 7501-7530 by the corresponding Init Data Change Segment Descriptor (the correspondence is given by the order) among those having the FIM ID field set to "FIM27 Segment Descriptor". Splitting of a segment of init data between this message and FIM28, Short Initialization Data Change 2 FIM, is not allowed.

28. FIM28: Short Initialization Data Change 2 FIM

This FIM is used to convey to the Terminal up to 15 segments (contiguous 16-bit data words) of initialization data. The position of each segment in the Init Data File is defined in Maintenance Parameters Data Words 7501-7530 by the corresponding Init Data Change Segment Descriptor (the correspondence is given by the order) among those having the FIM ID field set to "FIM28 Segment Descriptor". Splitting of a segment of init data between this message and FIM27, Short Initialization Data Change 1 FIM, is not allowed.

29. FIM29: Host Navigation Data Type 10 FIM

This FIM shall be used to convey the Inertial Navigation (IN) Data to the Terminal.

30. FIM30: Host Navigation Data Type 11 FIM

This message shall be used to convey Non-Inertial Navigation data (SAHR+ADC data) to the Terminal in case of loss of the information coming from the Inertial Navigation. Together with the Navigation data, it is necessary to supply also Fixing Data in order to allow the MIDS to restart after the Navigation reset.

31. FIM31: Host Navigation Data Type 12 FIM

This message shall be used to convey the Host Navigation Data (FIX) to the MIDS terminal. This FIM should be used in the following cases: To provide GPS update to the Terminal, to provide manual FIX update on pilot action, and to provide FIX update on NAV mode change.

32. FIM32: Host Navigation Data Type 13 FIM

This message shall be used to convey the Inertial Navigation Data.

33. FIM33: Host Navigation Data Type 14 FIM

This FIM shall be used to convey the Non Inertial Navigation Data to the MIDS terminal in case of loss of the information coming from the Inertial Navigation Unit.

34. FIM34: Short RC Response FIM

This FIM shall be used by the Host to convey to the Terminal the short form of a Host response to an R/C message received over the network.

VI. MIDS UNIVERSAL INTERFACE TESTING

The MIDS Universal interface is tested for two different platforms. For testing purposes, I have have created two platforms "4" and "10". For each platform, I have created a set of FIM messages that are unique to each platform. The messages have been selected using the MIDS Universal Interface Control Panel.

A. TEST CASE DESCRIPTION

1. Platform "4" Test Description

- Select platform “4” from the select platform window.
- Select FIM01, FIM02, FIM03, FIM10 and FIM15 from the select message window

Press the Code generator key from the MIDS Universal Selection Panel and verify that the following FimTemplate.java file is created in appendix B.

2. Platform "10" Test Description

- Select platform “10” from the select platform window.
- Select FIM01, FIM03, and FIM15 from the select message window
- Press the Code generator key from the MIDS Universal Selection Panel and verify that the following FimTemplate.java file is created in appendix C.

3. Test Limitations

In to order to verify that the program works properly, I need the hardware to verify that the messages are generated correctly, since I do not have the 1553 board to check the results of the messages transmitted or received. The amounts of software interfaces are limited to two test cases due to the potential number of interfaces that can be created using this approach.

My tests consist of a buffer that keeps track of the messages that have been selected and the messages and the message contents. The framework is tested as a standalone feature and are not be integrated with the other CSCIs in the MIDS terminal for the time being.

VII. MIDS TIO CONCLUSIONS AND FUTURE WORK

A. CONCLUSION/FUTURE WORK

I have demonstrated via rapid prototyping that the framework is capable of supporting the development of future interfaces (refer to paragraph E, Section III, page 26). The new framework gives us the flexibility that we do not currently have and that we need in order to develop interfaces at a low cost with a more manageable software development efforts.

The use of this framework saves time and money in the creation of future platforms. With the new framework platforms can be created or reconfigure easier and testing does not have to be very extensive after the software has been qualified once. The code can be generated automatically [ref. 3] to meet this requirement. This innovative approach definitely benefits this family of products.

Using generative programming allow us to use software building block to facilitate the development of software. The created objects have an identity and behavior associated with them. Any future modification of the MIDS interface software is easier to implement and diminishes the type of problems that we are currently experiencing.

Future work should include the hosting of the new framework into the Tailor lamina and run and test the new software in this environment. It is also important to do the integration between the new TIO framework and the other MIDS CSCIs.

The processing of these messages has not been tested for this framework or the interface to other CSCIs. The test cannot be done with the software hosted in an embedded system; additional work should be performed in this area to determine if any time constraints exist when this framework is run on the hardware.

B. RECOMMENDATION

This specification based design and the new framework can benefit the MIDS program if they are implemented. This approach can save time and money for the Program Office, this approach allows the flexibility needed to support future requirements. Currently, the Program Office does not have this capability built in the MIDS terminal and has to develop new interfaces. This framework should be integrated with the current TIO CSCI architecture to show that this is a viable approach. Additional studies should be done in this particular area to prove that this methodology can be used by the MIDS terminal and can benefit NATO and countries that determine that they want to use the MIDS terminal in their platforms.

Finally, this thesis should be available to all potential users of the MIDS terminal.

LIST OF REFERENCES

[1] MIL-STD-1553B Aircraft Internal Time Division Command/Response Multiplex Data Bus.

[2] MIDS Interface Control Document BAE Systems

[3] Generative Prrogramming Methods, Tools, and Applications Krzysztof Czarnecki and Ulrich W. Eisenecker

[4] Report by Software Engineering Group Naval Postgraduate School, Professor Berzins 1 October 1999.

[5] Software Architecture for Product Families: Principles and Practice, Mehdi Jazayeri, Alexander Ran, Frank van der Linden. Addison Wesley Longman, 2000.

[6] Testing Object-Oriented Systems Models, Patterns and Tools Robert V. Binder.

[7] Gamma E., Helm R., Johnson R., Vlissides J., *Design Patterns Elements of Reusable Object-Oriented Software*, Addison-Wesley 1995.

[8] Object Model driven Code Generation for the Enterprise William J Ray and Andy Farrar.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDICES

APPENDIX A- MIDS INTERFACE SOURCE CODE

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import java.io.*;
class Mids_interface_rev7
{
private static final String INDENT = "  ";
private static final String RETURN = "\r\n";
public static void main(String[] args)
        {
         JFrame f = new JFrame("   MIDS Interface Message Control Panel v1.0");
         f.setSize(400, 400);
         f.setLocation(200,100);
         f.addWindowListener(new WindowAdapter()
         {
           public void windowClosing(WindowEvent we)
           {
                   System.exit(0);
           }
         });
 //initialize the fim sequence
```

```
//Added to control the variable switch
// creates a combo box for selecting the platform number
String [] platforms = {"  00", "  01", "  02", "  03", "  04", "  05", "
06", "  07  ", "  08",
        "  09", "  10", "  11", "  12", "  13", "  14", "  15", "  16", "
17",
        "  18", "  19", "  20", "  21", "  22", "  23", "  24", "  25", "
26",
        "  27", "  28", "  29", "  30", "  31"};
final JComboBox comboBox = new JComboBox(platforms);
comboBox.setEditable(false);
comboBox.setSelectedIndex(5);
// create a combo box for FIM messages, these messages will be part of the
platform
//selection
String [] fims = {"FIM01","FIM02", "FIM03", "FIM04", "FIM05", "FIM06",
"FIM07", "FIM08", "FIM09", "FIM10", "FIM11", "FIM12", "FIM13", "FIM14",
"FIM15", "FIM16", "FIM17", "FIM18", "FIM19", "FIM20", "FIM21", "FIM22",
"FIM23", "FIM24", "FIM25", "FIM26", "FIM27", "FIM28", "FIM29", "FIM30",
"FIM31", "FIM32", "FIM33", "FIM34"};

JComboBox comboBox1 = new JComboBox(fims);
comboBox1.setEditable(true);
//create a list with the same data model for the platforms and
//FIMS
final JList list = new JList(comboBox.getModel());
```

```
final JList fimlist = new JList(comboBox1.getModel());
fimlist.setSelectedIndex(5);
 //Assigns buttons to select platform and messages
JLabel plat_label = new JLabel ("     Select Platform types     ");
JButton button_message = new JButton("     Select messages     ");

//The following code allows the selection of messages
//adds the listeners for the "Select messages" button
//"Select messages" button
button_message.addActionListener(new ActionListener()
 {
      public void actionPerformed(ActionEvent ae)
          {
      Object [] selectionfim = fimlist.getSelectedValues();
                System.out.println("------");
                        for (int i=0; i < selectionfim.length; i++)
                System.out.println(selectionfim[i]);
             }
    });
JPanel generatePanel = new JPanel(  );
JButton  generateButton = new JButton("Generate code");
generatePanel.add(generateButton);
generateButton.addActionListener(new ActionListener(  )
   {
     public void actionPerformed(ActionEvent ae)
```

```
                    {
                        System.out.println("code is associated with the generate code key");
                //code associate with the type of messages (FIMS)
                        Object [] selection = comboBox.getSelectedObjects();
                       System.out.println("------");
                        //number of selections in platform types or plat ID,
                   for (int i=0; i < selection.length; i++)
                   System.out.println("platform " +selection[i] +" messages");
                   System.out.println("platform selected values " +selection.length);
                   //code associated the selection of the FIM messages
                           Object [] selectionfim = fimlist.getSelectedValues();
                           // Call another function with the selected values
                            processData(selection[0], selectionfim);
                   System.out.println("------");
                                   for (int i=0; i < selectionfim.length; i++)
                                  System.out.println(selectionfim[i]);
                                   System.out.println("Select generate code button
processing");
                           }  // end of Action Performed
               });  // generateButton.addActionListener
               //put the controls in the panel
           java.awt.Container c = f.getContentPane();
           JPanel comboPanel = new JPanel();
            JPanel comboPanel1 = new JPanel();
           // Create a layout manager
            FlowLayout flow = new FlowLayout(FlowLayout.LEFT);
```

```
 c.setLayout(flow);
c.add(plat_label);
comboPanel.add(comboBox);
comboPanel1.add(comboBox1);
c.add(comboPanel);
c.add(button_message);
c.add(new JScrollPane(fimlist));
c.add(generatePanel);
f.setVisible(true);
  }  // end of main
  public static void processData(Object platform, Object []fims)
  {
    writeClassToFile(fims);
  }

private static void writeClassToFile(Object []fims) {
File destFile = new File("FimTemplate.java");
    try {
      // need to delete the old file before creating a new
      // template file
      destFile.delete();
      boolean haveLock = destFile.createNewFile();
      RandomAccessFile randomFile = new RandomAccessFile(destFile, "rw");
      randomFile.writeBytes("import java.awt.*;" + RETURN);
      randomFile.writeBytes("import java.awt.event.*;" + RETURN);
```

```
randomFile.writeBytes("import javax.swing.*;" + RETURN);
randomFile.writeBytes("import java.io.*;" + RETURN);
randomFile.writeBytes(RETURN);
randomFile.writeBytes("class FimTemplate {" + RETURN);
// write out writeData function
randomFile.writeBytes(INDENT + "private static void writeData(Fim []fims) {" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "File destFile = new File(\"Test.dat\");" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "try {" + RETURN);
randomFile.writeBytes(INDENT +INDENT + INDENT + "// Need to delete the old file before writing the new one" + RETURN);
randomFile.writeBytes(INDENT +INDENT + INDENT + "destFile.delete();" + RETURN);
randomFile.writeBytes(INDENT +INDENT + INDENT + "boolean haveLock = destFile.createNewFile();" + RETURN);
randomFile.writeBytes(INDENT +INDENT + INDENT + "RandomAccessFile randomFile = new RandomAccessFile(destFile, \"rw\");" + RETURN);
randomFile.writeBytes(INDENT +INDENT + INDENT + "for (int i = 0; i < fims.length; i++) {" + RETURN);
randomFile.writeBytes(INDENT + INDENT +INDENT + INDENT + "if (fims[i].getName().length() > 0) {" + RETURN);
randomFile.writeBytes(INDENT + INDENT + INDENT +INDENT + INDENT + "randomFile.writeBytes(fims[i].getName());  // write the name to the file" + RETURN);
randomFile.writeBytes(INDENT + INDENT + INDENT +INDENT + INDENT + "randomFile.writeBytes(\"\\r\\n\");" + RETURN);
```

```
randomFile.writeBytes(INDENT + INDENT + INDENT +INDENT + INDENT +
"randomFile.writeShort(fims[i].getWord0());" + RETURN);
randomFile.writeBytes(INDENT + INDENT + INDENT +INDENT + INDENT +
"randomFile.writeShort(fims[i].getWord1());" + RETURN);
randomFile.writeBytes(INDENT + INDENT + INDENT +INDENT + INDENT +
"randomFile.writeShort(fims[i].getWord2());" + RETURN);
randomFile.writeBytes(INDENT + INDENT + INDENT +INDENT + INDENT +
"randomFile.writeShort(fims[i].getWord3());" + RETURN);
randomFile.writeBytes(INDENT + INDENT + INDENT +INDENT + INDENT +
"randomFile.writeBytes(\"\\r\\n\");" + RETURN);
randomFile.writeBytes(INDENT + INDENT +INDENT + INDENT + "}" +
RETURN);
randomFile.writeBytes(INDENT +INDENT + INDENT + "}" + RETURN);
randomFile.writeBytes(INDENT +INDENT + INDENT + "randomFile.close();
// Done with the file so close it" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "}" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "catch (Exception e) {" +
RETURN);
randomFile.writeBytes(INDENT +INDENT + INDENT + "e.printStackTrace();"
+ RETURN);
randomFile.writeBytes(INDENT + INDENT + "}" + RETURN);
randomFile.writeBytes(INDENT + "}" + RETURN);
randomFile.writeBytes("\r\n");
// Now write out the main function
randomFile.writeBytes(INDENT + "public static void main(String args[]) {" +
RETURN);
```

```
        randomFile.writeBytes(INDENT + INDENT + "Fim []fimClasses = new Fim[" +
fims.length + "];" + RETURN);
            for (int i = 0; i < fims.length; i++) {
             if (i < 10) {
                        randomFile.writeBytes(INDENT + INDENT + "fimClasses[" + i
+ "] = new Fim0" + (i + 1) + "();" + RETURN);
              }
              else {
                    randomFile.writeBytes(INDENT + INDENT + "fimClasses[" + i + "] =
new Fim" + (i + 1) + "();" + RETURN);
              }
        randomFile.writeBytes(INDENT + INDENT + "fimClasses[" + i +
"].buildWord0();" + RETURN);
        randomFile.writeBytes(INDENT + INDENT + "fimClasses[" + i +
"].buildWord1();" + RETURN);
        randomFile.writeBytes(INDENT + INDENT + "fimClasses[" + i +
"].buildWord2();" + RETURN);
           randomFile.writeBytes(RETURN);
            }
        randomFile.writeBytes(INDENT + INDENT + "writeData(fimClasses);" +
RETURN);
        randomFile.writeBytes(INDENT + "}" + RETURN);
        randomFile.writeBytes("}" + RETURN);
        randomFile.writeBytes("abstract class Fim {" + RETURN);
        randomFile.writeBytes(INDENT + "private String name;" + RETURN);
        randomFile.writeBytes(INDENT + "private short word0;" + RETURN);
```

```
        randomFile.writeBytes(INDENT + "private short word1;" + RETURN);
        randomFile.writeBytes(INDENT + "private short word2;" + RETURN);
        randomFile.writeBytes(INDENT + "private short word3;" + RETURN);
        randomFile.writeBytes(RETURN);
        randomFile.writeBytes(INDENT + "// default constructor" + RETURN);
        randomFile.writeBytes(INDENT + "public Fim() {" + RETURN);
        randomFile.writeBytes(INDENT + INDENT + "// initialize the data" + RETURN);
        randomFile.writeBytes(INDENT + INDENT + "name = \"\";" + RETURN);
        randomFile.writeBytes(INDENT + INDENT + "word0 = 0;" + RETURN);
        randomFile.writeBytes(INDENT + INDENT + "word1 = 0;" + RETURN);
        randomFile.writeBytes(INDENT + INDENT + "word2 = 0;" + RETURN);
        randomFile.writeBytes(INDENT + INDENT + "word3 = 0;" + RETURN);
        randomFile.writeBytes(INDENT + "}" + RETURN);
        randomFile.writeBytes(RETURN);
        // Now write the non-abstract methods
        // These methods will just return the private data, so no need to make
        // them abstract
        randomFile.writeBytes(INDENT + "public final String getName() {" + RETURN);
        randomFile.writeBytes(INDENT + INDENT + "return name;" + RETURN);
        randomFile.writeBytes(INDENT + "}" + RETURN);
        randomFile.writeBytes(RETURN);
        randomFile.writeBytes(INDENT + "public final void setName(String newName) {" + RETURN);
        randomFile.writeBytes(INDENT + INDENT + "name = newName;" + RETURN);
```

```
        randomFile.writeBytes(INDENT + "}" + RETURN);
        randomFile.writeBytes(RETURN);
        randomFile.writeBytes(INDENT + "public final short getWord0() {" +
RETURN);
        randomFile.writeBytes(INDENT + INDENT + "return word0;" + RETURN);
        randomFile.writeBytes(INDENT + "}" + RETURN);
        randomFile.writeBytes(RETURN);
        randomFile.writeBytes(INDENT + "public final short getWord1() {" +
RETURN);
        randomFile.writeBytes(INDENT + INDENT + "return word1;" + RETURN);
        randomFile.writeBytes(INDENT + "}" + RETURN);
        randomFile.writeBytes(RETURN);
        randomFile.writeBytes(INDENT + "public final short getWord2() {" +
RETURN);
        randomFile.writeBytes(INDENT + INDENT + "return word2;" + RETURN);
        randomFile.writeBytes(INDENT + "}" + RETURN);
        randomFile.writeBytes(RETURN);
        randomFile.writeBytes(INDENT + "public final short getWord3() {" +
RETURN);
        randomFile.writeBytes(INDENT + INDENT + "return word3;" + RETURN);
        randomFile.writeBytes(INDENT + "}" + RETURN);
        randomFile.writeBytes(RETURN);
        randomFile.writeBytes(INDENT + "public final void setWord0(short newWord)
{" + RETURN);
        randomFile.writeBytes(INDENT + INDENT + "word0 = newWord;" +
RETURN);
```

```
        randomFile.writeBytes(INDENT + "}" + RETURN);
        randomFile.writeBytes(RETURN);
        randomFile.writeBytes(INDENT + "public final void setWord1(short newWord)
{" + RETURN);
        randomFile.writeBytes(INDENT + INDENT + "word1 = newWord;" +
RETURN);
        randomFile.writeBytes(INDENT + "}" + RETURN);
        randomFile.writeBytes(RETURN);
        randomFile.writeBytes(INDENT + "public final void setWord2(short newWord)
{" + RETURN);
        randomFile.writeBytes(INDENT + INDENT + "word2 = newWord;" +
RETURN);
        randomFile.writeBytes(INDENT + "}" + RETURN);
        randomFile.writeBytes(RETURN);
        randomFile.writeBytes(INDENT + "public final void setWord3(short newWord)
{" + RETURN);
        randomFile.writeBytes(INDENT + INDENT + "word3 = newWord;" +
RETURN);
        randomFile.writeBytes(INDENT + "}" + RETURN);
        randomFile.writeBytes(RETURN);
        // Now create the abstract methods that must be implemented in
        //  sub classes
        randomFile.writeBytes(INDENT + "abstract void buildWord0();" + RETURN);
        randomFile.writeBytes(INDENT + "abstract void buildWord1();" + RETURN);
        randomFile.writeBytes(INDENT + "abstract void buildWord2();" + RETURN);
        randomFile.writeBytes(INDENT + "abstract void buildWord3();" + RETURN);
```

```
        randomFile.writeBytes("}" + RETURN);
        randomFile.writeBytes(RETURN);
          for (int i = 0; i < fims.length; i++) {
              if (fims[i] == "FIM01") {
                randomFile.writeBytes("class Fim01 extends Fim {" + RETURN);
                randomFile.writeBytes(INDENT + "short tempWord;" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "public Fim01() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setName(\"FIM01\");" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord0() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "int carrierMessageId =5;" +
RETURN);
               randomFile.writeBytes(INDENT + INDENT + "int rcmessageId     =2;" +
RETURN);
              randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(rcmessageId << 10);" + RETURN);
              randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord | carrierMessageId);" + RETURN);
               randomFile.writeBytes(INDENT + INDENT + "setWord0(tempWord);" +
RETURN);
               randomFile.writeBytes(INDENT + "}" + RETURN);
               randomFile.writeBytes(RETURN);
               randomFile.writeBytes(INDENT + "void buildWord1() {" + RETURN);
```

```
                randomFile.writeBytes(INDENT + INDENT + "int type           =1;" +
RETURN);
                randomFile.writeBytes(INDENT + INDENT + "int ijms_Select   =1;" +
RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "int packingLimit =4;" +
RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "int length        =3;" +
RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "int priority      =3;" +
RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(priority <<11);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord | (length <<3));" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord | (packingLimit<<5));" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord | (ijms_Select <<2));" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord |  type);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord1(tempWord);" +
RETURN);
               randomFile.writeBytes(INDENT + "}" + RETURN);
               randomFile.writeBytes(RETURN);
               randomFile.writeBytes(INDENT + "void buildWord2() {" + RETURN);
               randomFile.writeBytes(INDENT + INDENT + "int timeTagMSW   =10;" +
RETURN);
```

```
                randomFile.writeBytes(INDENT + INDENT + "int timeTagLSW   =10;" +
RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(timeTagMSW<<16);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord | timeTagLSW <<16);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord2(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord3() {" + RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes("}" + RETURN);
               } // end of if statement FIM01
               if (fims[i] =="FIM02") {
                randomFile.writeBytes("class Fim02 extends Fim {" + RETURN);
                randomFile.writeBytes(INDENT + "short tempWord;" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "public Fim02() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setName(\"FIM02\");" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord0() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "int requestNumber = 10;" +
RETURN);
```

```
                randomFile.writeBytes(INDENT + INDENT + "int requestType = 5;" +
RETURN);
               randomFile.writeBytes(INDENT + INDENT + "int dataWordCount = 7;" +
RETURN);
                  randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(requestNumber << 8);" + RETURN);
                  randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord |  (requestType << 5));" + RETURN);
                  randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord | dataWordCount);" + RETURN);
                   randomFile.writeBytes(INDENT + INDENT + "setWord0(tempWord);"
+ RETURN);
                   randomFile.writeBytes(INDENT + "}" + RETURN);
                   randomFile.writeBytes(RETURN);
                   randomFile.writeBytes(INDENT + "void buildWord1() {" + RETURN);
                  randomFile.writeBytes(INDENT + INDENT + "int
coreMemoryAddressMSW = 31;" + RETURN);
                  randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(coreMemoryAddressMSW);" + RETURN);
                  randomFile.writeBytes(INDENT + INDENT + "setWord1(tempWord);"
+ RETURN);
                  randomFile.writeBytes(INDENT + "}" + RETURN);
                  randomFile.writeBytes(RETURN);
                  randomFile.writeBytes(INDENT + "void buildWord2() {" + RETURN);
                  randomFile.writeBytes(INDENT + INDENT + "int
coreMemoryAddressLSW = 10;" + RETURN);
```

```
                    randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(coreMemoryAddressLSW);" + RETURN);
                    randomFile.writeBytes(INDENT + INDENT + "setWord2(tempWord);"
+ RETURN);
                    randomFile.writeBytes(INDENT + "}" + RETURN);
                    randomFile.writeBytes(RETURN);
                    randomFile.writeBytes(INDENT + "void buildWord3() {" + RETURN);
                   randomFile.writeBytes(INDENT + INDENT + "setWord3(tempWord);" +
RETURN);
                    randomFile.writeBytes(INDENT + "}" + RETURN);
                    randomFile.writeBytes(RETURN);
                   randomFile.writeBytes("}" + RETURN);
                } // end of if FIM02 statement
                if (fims[i] =="FIM03") {
                  randomFile.writeBytes("class Fim03 extends Fim {" + RETURN);
                  randomFile.writeBytes(INDENT + "short tempWord;" + RETURN);
                  randomFile.writeBytes(RETURN);
                  randomFile.writeBytes(INDENT + "public Fim03() {" + RETURN);
                  randomFile.writeBytes(INDENT + INDENT + "setName(\"FIM03\");" +
RETURN);
                  randomFile.writeBytes(INDENT + "}" + RETURN);
                  randomFile.writeBytes(RETURN);
                  randomFile.writeBytes(INDENT + "void buildWord0() {" + RETURN);
                  randomFile.writeBytes(INDENT + INDENT + "int setIndexNumber = 1;"
+ RETURN);
```

```
                randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(setIndexNumber << 4);" + RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "setWord0(tempWord);" + RETURN);
                 randomFile.writeBytes(INDENT + "}" + RETURN);
                 randomFile.writeBytes(RETURN);
                 randomFile.writeBytes(INDENT + "void buildWord1() {" + RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "int transferSegmentCount = 30;" + RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(transferSegmentCount);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord1(tempWord);" + RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord2() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord3(tempWord);" + RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord3() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord3(tempWord);" + RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                 randomFile.writeBytes("}" + RETURN);
```

```
            } // end of if FIM03 statement
            if (fims[i] =="FIM04") {
              randomFile.writeBytes("class Fim04 extends Fim {" + RETURN);
              randomFile.writeBytes(INDENT + "short tempWord;" + RETURN);
              randomFile.writeBytes(RETURN);
              randomFile.writeBytes(INDENT + "public Fim04() {" + RETURN);
              randomFile.writeBytes(INDENT + INDENT + "setName(\"FIM04\");" +
RETURN);
              randomFile.writeBytes(INDENT + "}" + RETURN);
              randomFile.writeBytes(RETURN);
              randomFile.writeBytes(INDENT + "void buildWord0() {" + RETURN);
               randomFile.writeBytes(INDENT + INDENT + "int TV = 1;" + RETURN);
              randomFile.writeBytes(INDENT + INDENT + "int hrsTensDigits = 2;" +
RETURN);
              randomFile.writeBytes(INDENT + INDENT + "int hrsOnesDigits = 1;" +
RETURN);
              randomFile.writeBytes(INDENT + INDENT + "tempWord =  (short)(TV
<< 14);" + RETURN);
              randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord |  (hrsTensDigits <<3));" + RETURN);
              randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord | hrsOnesDigits);" + RETURN);
               randomFile.writeBytes(INDENT + INDENT + "setWord0(tempWord);" +
RETURN);
              randomFile.writeBytes(INDENT + "}" + RETURN);
              randomFile.writeBytes(RETURN);
```

```
randomFile.writeBytes(INDENT + "void buildWord1() {" + RETURN);
 randomFile.writeBytes(INDENT + INDENT + "int minutesTensDigits =2;" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "int minutesOnesDigits =5;" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "int secondsTensDigits =50;" + RETURN);
 randomFile.writeBytes(INDENT + INDENT + "int secondsOnesTensDigits =5;" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(minutesTensDigits <<11);" + RETURN);
 randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord | minutesOnesDigits <<7);" + RETURN);
 randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord |  (secondsOnesTensDigits <<3));" + RETURN);
 randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord | secondsOnesTensDigits);" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "setWord1(tempWord);" + RETURN);
randomFile.writeBytes(INDENT + "}" + RETURN);
randomFile.writeBytes(RETURN);
randomFile.writeBytes(INDENT + "void buildWord2() {" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "int yearTensDigit = 01;" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "int yearOnesDigit =2;" + RETURN);
```

```
                randomFile.writeBytes(INDENT + INDENT + "int timeFigureOfMerit =
5;" + RETURN);
                        randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord | yearTensDigit <<11);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord |  (yearOnesDigit << 7));" + RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord | timeFigureOfMerit);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord2(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord3() {" + RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "setWord3(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes("}" + RETURN);
               } // end of if FIM04 statement
               //begining of FIM 05
               if (fims[i] =="FIM05") {
                randomFile.writeBytes("class Fim05 extends Fim {" + RETURN);
                randomFile.writeBytes(INDENT + "short tempWord;" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "public Fim05() {" + RETURN);
```

```
                randomFile.writeBytes(INDENT + INDENT + "setName(\"FIM05\");" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord0() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "int addressTrackNumber =
7;" + RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(addressTrackNumber = 7);" + RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "setWord0(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                 randomFile.writeBytes(INDENT + "void buildWord1() {" + RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "setWord1(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord2() {" + RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "setWord2(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord3() {" + RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "setWord3(tempWord);" +
RETURN);
```

```
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes("}" + RETURN);
                 } // end of if FIM05 statement
               //Beginning of FIM06

              if (fims[i] =="FIM06") {
                randomFile.writeBytes("class Fim06 extends Fim {" + RETURN);
                randomFile.writeBytes(INDENT + "short tempWord;" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "public Fim06() {" + RETURN);
                  randomFile.writeBytes(INDENT + INDENT + "setName(\"FIM06\");" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                  randomFile.writeBytes(INDENT + "void buildWord0() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "int messageRTTAId =10;"
+ RETURN);
                        randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(messageRTTAId =10);" + RETURN);
                  randomFile.writeBytes(INDENT + INDENT + "setWord0(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                  randomFile.writeBytes(INDENT + "void buildWord1() {" + RETURN);
```

```
                randomFile.writeBytes(INDENT + INDENT + "int
addresseeTrackNumber = 12;" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(addresseeTrackNumber);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord1(tempWord);" +
RETURN);
               randomFile.writeBytes(INDENT + "}" + RETURN);
               randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord2() {" + RETURN);
               randomFile.writeBytes(INDENT + INDENT + "setWord2(tempWord);" +
RETURN);
               randomFile.writeBytes(INDENT + "}" + RETURN);
               randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord3() {" + RETURN);
               randomFile.writeBytes(INDENT + INDENT + "setWord3(tempWord);" +
RETURN);
               randomFile.writeBytes(INDENT + "}" + RETURN);
               randomFile.writeBytes(RETURN);
                randomFile.writeBytes("}" + RETURN);
                 } // end of if FIM06 statement
              //Beginning FIM07
              if (fims[i] =="FIM07") {
               randomFile.writeBytes("class Fim07 extends Fim {" + RETURN);
               randomFile.writeBytes(INDENT + "short tempWord;" + RETURN);
               randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "public Fim07() {" + RETURN);
```

```
randomFile.writeBytes(INDENT + INDENT + "setName(\"FIM07\");" +
RETURN);
randomFile.writeBytes(INDENT + "}" + RETURN);
randomFile.writeBytes(RETURN);
 randomFile.writeBytes(INDENT + "void buildWord0() {" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "int reductionMessageId =
25;" + RETURN);
 randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(reductionMessageId = 25);" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "setWord0(tempWord);" +
RETURN);
randomFile.writeBytes(INDENT + "}" + RETURN);
randomFile.writeBytes(RETURN);
randomFile.writeBytes(INDENT + "void buildWord1() {" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "int reductionPackingLimit
= 10;" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "int reductionLength = 3;" +
RETURN);
randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(reductionPackingLimit <<7);" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord |  (reductionLength));" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "setWord1(tempWord);" +
RETURN);
randomFile.writeBytes(INDENT + "}" + RETURN);
randomFile.writeBytes(RETURN);
```

```
                randomFile.writeBytes(INDENT + "void buildWord2() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "int reductionTimeTagMSW
=20;" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(reductionTimeTagMSW);" + RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "setWord2(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord3() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "int reductionTimeTagLSW
=50;" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(reductionTimeTagLSW);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord3(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes("}" + RETURN);
              } // end of if FIM07 statement
              //begining FIM08
              if (fims[i] =="FIM08") {
                 randomFile.writeBytes("class Fim08 extends Fim {" + RETURN);
                 randomFile.writeBytes(INDENT + "short tempWord;" + RETURN);
                 randomFile.writeBytes(RETURN);
                 randomFile.writeBytes(INDENT + "public Fim08() {" + RETURN);
```

```
                randomFile.writeBytes(INDENT + INDENT + "setName(\"FIM08\");" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord0() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "int routeEstablishmentId
=10;" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(routeEstablishmentId =10);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord0(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord1() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "int
routeEstablishmentTrackNumber =7;" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(routeEstablishmentTrackNumber);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord1(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord2() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "int
routeEstablishmentAddresseeTrackNumber = 25;" + RETURN);
```

```
                randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(routeEstablishmentAddresseeTrackNumber);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord2(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord3() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord3(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes("}" + RETURN);

            } // end of if FIM08 statement
            //begining of FIM09
            if (fims[i] =="FIM09") {
              randomFile.writeBytes("class Fim09 extends Fim {" + RETURN);
              randomFile.writeBytes(INDENT + "short tempWord;" + RETURN);
              randomFile.writeBytes(RETURN);
              randomFile.writeBytes(INDENT + "public Fim09() {" + RETURN);
              randomFile.writeBytes(INDENT + INDENT + "setName(\"FIM09\");" +
RETURN);
              randomFile.writeBytes(INDENT + "}" + RETURN);
              randomFile.writeBytes(RETURN);
              randomFile.writeBytes(INDENT + "void buildWord0() {" + RETURN);
```

```
randomFile.writeBytes(INDENT + INDENT + "int VMFMessageId =25;" +
RETURN);
randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(VMFMessageId );" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "setWord0(tempWord);" +
RETURN);
randomFile.writeBytes(INDENT + "}" + RETURN);
randomFile.writeBytes(RETURN);
randomFile.writeBytes(INDENT + "void buildWord1() {" + RETURN);
 randomFile.writeBytes(INDENT + INDENT + "int
VMFmessageXmCounter =33;" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(VMFmessageXmCounter);" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "setWord1(tempWord);" +
RETURN);
randomFile.writeBytes(INDENT + "}" + RETURN);
randomFile.writeBytes(RETURN);
randomFile.writeBytes(INDENT + "void buildWord2() {" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "int
VMFmessageRcvCounter =30;" + RETURN);
 randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(VMFmessageRcvCounter);" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "setWord2(tempWord);" +
RETURN);
randomFile.writeBytes(INDENT + "}" + RETURN);
randomFile.writeBytes(RETURN);
```

```
                randomFile.writeBytes(INDENT + "void buildWord3() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord3(tempWord);" +
RETURN);
                 randomFile.writeBytes(INDENT + "}" + RETURN);
                 randomFile.writeBytes(RETURN);
                 randomFile.writeBytes("}" + RETURN);

               } // end of if FIM09 statement
               //begining FIM10
               if (fims[i] =="FIM10") {
                 randomFile.writeBytes("class Fim10 extends Fim {" + RETURN);
                 randomFile.writeBytes(INDENT + "short tempWord;" + RETURN);
                 randomFile.writeBytes(RETURN);
                  randomFile.writeBytes(INDENT + "public Fim10() {" + RETURN);
                                  randomFile.writeBytes(INDENT + INDENT +
"setName(\"FIM10\");" + RETURN);
                 randomFile.writeBytes(INDENT + "}" + RETURN);
                 randomFile.writeBytes(RETURN);
                 randomFile.writeBytes(INDENT + "void buildWord0() {" + RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "int drv = 0;" + RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "int wav = 0;" + RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "int av = 0;" + RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "int hv = 0;" + RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "int hvv = 0;" + RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "int vvv = 0;" + RETURN);
```

```
            randomFile.writeBytes(INDENT + INDENT + "int insvcv = 0;" +
RETURN);
            randomFile.writeBytes(INDENT + INDENT + "int instcv = 0;" +
RETURN);
            randomFile.writeBytes(INDENT + INDENT + "int inertial = 0;" +
RETURN);
            randomFile.writeBytes(INDENT + INDENT + "int rdt = 0;" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "int rst = 0;" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "int gfv = 0;" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "int ains = 0;" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(drv <<
14);" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord |  wav <<13);" + RETURN);
             randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord |  av <<12);" + RETURN);
             randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord |  hv << 11);" + RETURN);
             randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord |  hvv <<10);" + RETURN);
             randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord |  vvv <<9);" + RETURN);
             randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord |  insvcv<<8);" + RETURN);
             randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord |  instcv<<7);" + RETURN);
```

```
                randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord |  inertial<<4);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord |  rdt <<2);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord |  rst <<1);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord |  gfv<<1);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord |  ains);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord0(tempWord);" + RETURN);
              randomFile.writeBytes(INDENT + "}" + RETURN);
              randomFile.writeBytes(RETURN);
               randomFile.writeBytes(INDENT + "void buildWord1() {" + RETURN);
              randomFile.writeBytes(INDENT + INDENT + "int gvv = 0;" + RETURN);
              randomFile.writeBytes(INDENT + INDENT + "int ghv = 0;" + RETURN);
              randomFile.writeBytes(INDENT + INDENT + "int gfvs= 0;" + RETURN);
              randomFile.writeBytes(INDENT + INDENT + "int velocitySource = 0;" + RETURN);
              randomFile.writeBytes(INDENT + INDENT + "int AIP = 0;" + RETURN);
              randomFile.writeBytes(INDENT + INDENT + "int INSAlignment = 0;" + RETURN);
              randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(gvv <<11);" + RETURN);
              randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord | ghv << 10);" + RETURN);
```

```
                randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord |  gfvs << 8);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord |  velocitySource << 4);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord |  AIP << 2);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord |  INSAlignment);" + RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "setWord1(tempWord);" + RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord2() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "int deadReckoningLatitude =100;" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(deadReckoningLatitude );" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord2(tempWord);" + RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);

                randomFile.writeBytes(INDENT + "void buildWord3() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "int deadReckoningLongitude =100;" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(deadReckoningLongitude );" + RETURN);
```

```
                randomFile.writeBytes(INDENT + INDENT + "setWord3(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes("}" + RETURN);
               } // end of if FIM10 statement
               //begining of FIM11
               if (fims[i] =="FIM11") {
                  randomFile.writeBytes("class Fim11 extends Fim {" + RETURN);
                  randomFile.writeBytes(INDENT + "short tempWord;" + RETURN);
                  randomFile.writeBytes(RETURN);
                  randomFile.writeBytes(INDENT + "public Fim11() {" + RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "setName(\"FIM11\");" +
RETURN);
                  randomFile.writeBytes(INDENT + "}" + RETURN);
                  randomFile.writeBytes(RETURN);
                  randomFile.writeBytes(INDENT + "void buildWord0() {" + RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "int externalTimeHrsDigit2
=10;" + RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "int
externalTimeHrsDigit1=20;" + RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(externalTimeHrsDigit2 << 10);" + RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord | externalTimeHrsDigit1);" + RETURN);
```

```
                randomFile.writeBytes(INDENT + INDENT + "setWord0(tempWord);" +
RETURN);
                 randomFile.writeBytes(INDENT + "}" + RETURN);
                 randomFile.writeBytes(RETURN);
                 randomFile.writeBytes(INDENT + "void buildWord1() {" + RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "int ExternalMinutesDigit2
= 100;" + RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "int ExternalMinutesDigit1
= 100;" + RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "int ExternalSecondsDigit2
= 100;" + RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "int ExternalSecondsDigit1
= 100;" + RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(ExternalMinutesDigit2 <<11);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord | ExternalSecondsDigit2<<3);" + RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord | ExternalSecondsDigit2<<3);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord | ExternalSecondsDigit1);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord1(tempWord);" +
RETURN);
                 randomFile.writeBytes(INDENT + "}" + RETURN);
                 randomFile.writeBytes(RETURN);
                 randomFile.writeBytes(INDENT + "void buildWord2() {" + RETURN);
```

```
                randomFile.writeBytes(INDENT + INDENT + "int dayOfYeardigit3 =20;"
+ RETURN);
               randomFile.writeBytes(INDENT + INDENT + "int dayOfYeardigit2 =
100;" + RETURN);
               randomFile.writeBytes(INDENT + INDENT + "int dayOfYeardigit1 =50;"
+ RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord2(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord3() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord3(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes("}" + RETURN);
              } // end of if FIM11 statement
              // begining of FIM 12
              if (fims[i] =="FIM12") {
                randomFile.writeBytes("class Fim12 extends Fim {" + RETURN);
                randomFile.writeBytes(INDENT + "short tempWord;" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "public Fim12() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setName(\"FIM12\");" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
```

```
            randomFile.writeBytes(RETURN);
            randomFile.writeBytes(INDENT + "void buildWord0() {" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "int xyChannel = 1;" +
RETURN);
            randomFile.writeBytes(INDENT + INDENT + "int tacanMode =1;" +
RETURN);
            randomFile.writeBytes(INDENT + INDENT + "int trRo =1;" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "int TacanAntenna =2;" +
RETURN);
            randomFile.writeBytes(INDENT + INDENT + "int TacanChannel = 8;" +
RETURN);
            randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(xyChannel << 13);" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord | tacanMode << 12);" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord | trRo<<11);" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord | TacanAntenna <<9);" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord | TacanChannel);" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "setWord0(tempWord);" +
RETURN);
            randomFile.writeBytes(INDENT + "}" + RETURN);
            randomFile.writeBytes(RETURN);
            randomFile.writeBytes(INDENT + "void buildWord1() {" + RETURN);
```

```
                randomFile.writeBytes(INDENT + INDENT + "setWord1(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord2() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord2(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord3() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord3(tempWord);" +
RETURN);
                  randomFile.writeBytes(INDENT + "}" + RETURN);
                  randomFile.writeBytes(RETURN);
                  randomFile.writeBytes("}" + RETURN);

               } // end of if FIM12 statement
              //Begining of FIM13
              if (fims[i] =="FIM13") {
                  randomFile.writeBytes("class Fim13 extends Fim {" + RETURN);
                  randomFile.writeBytes(INDENT + "short tempWord;" + RETURN);
                  randomFile.writeBytes(RETURN);
                  randomFile.writeBytes(INDENT + "public Fim13() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setName(\"FIM13\");" +
RETURN);
                  randomFile.writeBytes(INDENT + "}" + RETURN);
```

```
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord0() {" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "int TacanControlAntenna
=2;" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "int controlPower1 =1;" +
RETURN);
            randomFile.writeBytes(INDENT + INDENT + "int controlTacanMode =1;"
+ RETURN);
            randomFile.writeBytes(INDENT + INDENT + "int controlTrRo =1;" +
RETURN);
            randomFile.writeBytes(INDENT + INDENT + "int controlXY =1;" +
RETURN);
            randomFile.writeBytes(INDENT + INDENT + "int controlPower2=0;" +
RETURN);
            randomFile.writeBytes(INDENT + INDENT + "int controlTacanChannel
=5;" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(TacanControlAntenna << 13);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord | controlPower1 <<11);" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord | controlTacanMode <<10);" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord | controlTrRo <<9);" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord | controlXY <<8);" + RETURN);
```

```
                randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord | controlPower2 <<7);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord | controlTacanChannel);" + RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "setWord0(tempWord);" + RETURN);
                 randomFile.writeBytes(INDENT + "}" + RETURN);
                 randomFile.writeBytes(RETURN);
                 randomFile.writeBytes(INDENT + "void buildWord1() {" + RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "setWord1(tempWord);" + RETURN);
                 randomFile.writeBytes(INDENT + "}" + RETURN);
                 randomFile.writeBytes(RETURN);
                 randomFile.writeBytes(INDENT + "void buildWord2() {" + RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "setWord2(tempWord);" + RETURN);
                 randomFile.writeBytes(INDENT + "}" + RETURN);
                 randomFile.writeBytes(RETURN);
                 randomFile.writeBytes(INDENT + "void buildWord3() {" + RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "setWord3(tempWord);" + RETURN);
                 randomFile.writeBytes(INDENT + "}" + RETURN);
                 randomFile.writeBytes(RETURN);
                 randomFile.writeBytes("}" + RETURN);

             } // end of if FIM13 statement
```

```
            //Begining of FIM14 statement
            if (fims[i] =="FIM14") {
              randomFile.writeBytes("class Fim14 extends Fim {" + RETURN);
              randomFile.writeBytes(INDENT + "short tempWord;" + RETURN);
              randomFile.writeBytes(RETURN);
              randomFile.writeBytes(INDENT + "public Fim14() {" + RETURN);
              randomFile.writeBytes(INDENT + INDENT + "setName(\"FIM14\");" +
RETURN);
              randomFile.writeBytes(INDENT + "}" + RETURN);
              randomFile.writeBytes(RETURN);
              randomFile.writeBytes(INDENT + "void buildWord0() {" + RETURN);
              randomFile.writeBytes(INDENT + INDENT + "int PPLIMessageId =15;" +
RETURN);
              randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(PPLIMessageId);" + RETURN);
              randomFile.writeBytes(INDENT + INDENT + "setWord0(tempWord);" +
RETURN);
              randomFile.writeBytes(INDENT + "}" + RETURN);
              randomFile.writeBytes(RETURN);
              randomFile.writeBytes(INDENT + "void buildWord1() {" + RETURN);
              randomFile.writeBytes(INDENT + INDENT + "int PPLIPGxm =10;" +
RETURN);
              randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(PPLIPGxm);" + RETURN);
```

```
                randomFile.writeBytes(INDENT + INDENT + "setWord1(tempWord);" +
RETURN);
                  randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord2() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord2(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord3() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord3(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                  randomFile.writeBytes("}" + RETURN);
               } // end of if FIM14 statement
               //begining of Fim15
               if (fims[i] =="FIM15") {
                  randomFile.writeBytes("class Fim15 extends Fim {" + RETURN);
                  randomFile.writeBytes(INDENT + "short tempWord;" + RETURN);
                  randomFile.writeBytes(RETURN);
                  randomFile.writeBytes(INDENT + "public Fim15() {" + RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "setName(\"FIM15\");" +
RETURN);
                  randomFile.writeBytes(INDENT + "}" + RETURN);
                  randomFile.writeBytes(RETURN);
```

```
                randomFile.writeBytes(INDENT + "void buildWord0() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "int trackEntyityType =10;" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "int trackIndexNumber =15;" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "int trackNumberMSW =2;" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(trackEntyityType<<11);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord | trackIndexNumber <<5);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord | trackNumberMSW);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord0(tempWord);" + RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord1() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "int trackNumberLSW =15;" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(trackNumberLSW);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord1(tempWord);" + RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord2() {" + RETURN);
```

```
                randomFile.writeBytes(INDENT + INDENT + "setWord2(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord3() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord3(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes("}" + RETURN);

             } // end of if FIM15 statement
            //begining of FIM16
            if (fims[i] =="FIM16") {
              randomFile.writeBytes("class Fim16 extends Fim {" + RETURN);
              randomFile.writeBytes(INDENT + "short tempWord;" + RETURN);
              randomFile.writeBytes(RETURN);
              randomFile.writeBytes(INDENT + "public Fim16() {" + RETURN);
              randomFile.writeBytes(INDENT + INDENT + "setName(\"FIM16\");" +
RETURN);
              randomFile.writeBytes(INDENT + "}" + RETURN);
              randomFile.writeBytes(RETURN);
              randomFile.writeBytes(INDENT + "void buildWord0() {" + RETURN);
              randomFile.writeBytes(INDENT + INDENT + "int vvv =1;" + RETURN);
              randomFile.writeBytes(INDENT + INDENT + "int evv =0;" + RETURN);
              randomFile.writeBytes(INDENT + INDENT + "int nvv=0;" + RETURN);
```

```
            randomFile.writeBytes(INDENT + INDENT + "int alv =1;" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "int lov =0;" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "int lav =1;" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "tempWord =  (short)(vvv
<< 4);" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord | evv <<3);" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord | nvv <<2);" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord | alv <<1);" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord | lov <<8);" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord | lav);" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "setWord0(tempWord);" +
RETURN);
            randomFile.writeBytes(INDENT + "}" + RETURN);
            randomFile.writeBytes(RETURN);
            randomFile.writeBytes(INDENT + "void buildWord1() {" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "setWord1(tempWord);" +
RETURN);
            randomFile.writeBytes(INDENT + "}" + RETURN);
            randomFile.writeBytes(RETURN);
            randomFile.writeBytes(INDENT + "void buildWord2() {" + RETURN);
```

```
                randomFile.writeBytes(INDENT + INDENT + "setWord2(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord3() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord3(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes("}" + RETURN);

              } // end of if FIM16 statement
              //Begining of FIM 17
              if (fims[i] =="FIM17") {
                randomFile.writeBytes("class Fim17 extends Fim {" + RETURN);
                randomFile.writeBytes(INDENT + "short tempWord;" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "public Fim17() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setName(\"FIM17\");" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord0() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord0(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
```

```
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord1() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "int type3vvv = 2;" +
RETURN);
                randomFile.writeBytes(INDENT + INDENT + "int type3alv =1;" +
RETURN);
                randomFile.writeBytes(INDENT + INDENT + "int type3asv =0;;" +
RETURN);
                randomFile.writeBytes(INDENT + INDENT + "int type3atv =1;" +
RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(type3vvv << 3);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord | type3alv <<2);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord | type3asv <<2);" + RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord | type3atv);" + RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "setWord1(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord2() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord2(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
```

```
                randomFile.writeBytes(INDENT + "void buildWord3() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord3(tempWord);" +
RETURN);
                 randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);


                 randomFile.writeBytes("}" + RETURN);


               } // end of if FIM17 statement
               //begining of FIM18 Statement
               if (fims[i] =="FIM18") {
                randomFile.writeBytes("class Fim18 extends Fim {" + RETURN);
                randomFile.writeBytes(INDENT + "short tempWord;" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "public Fim18() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setName(\"FIM18\");" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord0() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "int type4vld = 1;" +
RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(type4vld);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord0(tempWord);" +
RETURN);
```

```
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord1() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord1(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord2() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord2(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord3() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord3(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes("}" + RETURN);

              } // end of if FIM18 statement
              //Begining of FIM19 message
              if (fims[i] =="FIM19") {
                randomFile.writeBytes("class Fim19 extends Fim {" + RETURN);
                randomFile.writeBytes(INDENT + "short tempWord;" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "public Fim19() {" + RETURN);
```

```
            randomFile.writeBytes(INDENT + INDENT + "setName(\"FIM19\");" +
RETURN);
            randomFile.writeBytes(INDENT + "}" + RETURN);
            randomFile.writeBytes(RETURN);
            randomFile.writeBytes(INDENT + "void buildWord0() {" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "setWord0(tempWord);" +
RETURN);
            randomFile.writeBytes(INDENT + "}" + RETURN);
            randomFile.writeBytes(RETURN);
            randomFile.writeBytes(INDENT + "void buildWord1() {" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "int type5alv =1;" +
RETURN);
            randomFile.writeBytes(INDENT + INDENT + "int type5pv = 0;" +
RETURN);
            randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(type5alv << 1);" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(type5pv);" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "setWord1(tempWord);" +
RETURN);
            randomFile.writeBytes(INDENT + "}" + RETURN);
            randomFile.writeBytes(RETURN);
            randomFile.writeBytes(INDENT + "void buildWord2() {" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "setWord2(tempWord);" +
RETURN);
            randomFile.writeBytes(INDENT + "}" + RETURN);
```

```
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord3() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord3(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                 randomFile.writeBytes("}" + RETURN);
               } // end of if FIM19 statement
               //Begining of FIM20 statement
               if (fims[i] =="FIM20") {
                randomFile.writeBytes("class Fim20 extends Fim {" + RETURN);
                randomFile.writeBytes(INDENT + "short tempWord;" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "public Fim20() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setName(\"FIM20\");" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord0() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "int type3XYmode = 1;" +
RETURN);
                randomFile.writeBytes(INDENT + INDENT + "int type3TrRo  = 0;" +
RETURN);
                randomFile.writeBytes(INDENT + INDENT + "int type3TacanMode =1;" +
RETURN);
```

```
            randomFile.writeBytes(INDENT + INDENT + "int type3tcv =1;" +
RETURN);
            randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(type3XYmode << 14);" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord | type3TrRo<<9);" + RETURN);
             randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord | type3TacanMode <<8);" + RETURN);
             randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord | type3tcv <<14);" + RETURN);


            randomFile.writeBytes(INDENT + INDENT + "setWord0(tempWord);" +
RETURN);
            randomFile.writeBytes(INDENT + "}" + RETURN);
            randomFile.writeBytes(RETURN);
            randomFile.writeBytes(INDENT + "void buildWord1() {" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "setWord1(tempWord);" +
RETURN);
            randomFile.writeBytes(INDENT + "}" + RETURN);
            randomFile.writeBytes(RETURN);
            randomFile.writeBytes(INDENT + "void buildWord2() {" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "setWord2(tempWord);" +
RETURN);
            randomFile.writeBytes(INDENT + "}" + RETURN);
            randomFile.writeBytes(RETURN);
```

```
                randomFile.writeBytes(INDENT + "void buildWord3() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord3(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes("}" + RETURN);
               } // end of if FIM20 statement
              //Begining FIM21
               if (fims[i] =="FIM21") {
                randomFile.writeBytes("class Fim21 extends Fim {" + RETURN);
                randomFile.writeBytes(INDENT + "short tempWord;" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "public Fim21() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setName(\"FIM21\");" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord0() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "int MLC =2;" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord =  (short)(MLC
<< 13);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord0(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord1() {" + RETURN);
```

```
                randomFile.writeBytes(INDENT + INDENT + "setWord1(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord2() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord2(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord3() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord3(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes("}" + RETURN);
               } // end of if FIM21 statement
               //begining of FIM22
               if (fims[i] =="FIM22") {
                randomFile.writeBytes("class Fim22 extends Fim {" + RETURN);
                randomFile.writeBytes(INDENT + "short tempWord;" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "public Fim22() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setName(\"FIM22\");" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
```

```
randomFile.writeBytes(INDENT + "void buildWord0() {" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "int type4ControlTacanAntenna = 2;" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "int type4ControlTacanChannelTens = 10;" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "int type4ControlTacanChannelUnits = 1;" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "int type4ControlTacanSt = 1;" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "int type4ControlTacanXY =0;" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "int type4ControlTacanMode =1;" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "int type4ControlTacanTrRo =0;" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(type4ControlTacanAntenna << 12);" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(type4ControlTacanChannelTens<< 4);" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord | type4ControlTacanSt <<2);" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord | type4ControlTacanMode<<1);" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord | type4ControlTacanTrRo);" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "setWord0(tempWord);" + RETURN);
```

```
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord1() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "int transferSegmentCount =
30;" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(transferSegmentCount);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord1(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);

                randomFile.writeBytes(INDENT + "void buildWord2() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord2(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord3() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord3(tempWord);" +
RETURN);
                 randomFile.writeBytes(INDENT + "}" + RETURN);
                 randomFile.writeBytes(RETURN);
                randomFile.writeBytes("}" + RETURN);
                //
               } // end of if FIM22 statement
               //begining of FIM23
```

```
            if (fims[i] =="FIM23") {
              randomFile.writeBytes("class Fim23 extends Fim {" + RETURN);
              randomFile.writeBytes(INDENT + "short tempWord;" + RETURN);
              randomFile.writeBytes(RETURN);
              randomFile.writeBytes(INDENT + "public Fim23() {" + RETURN);
              randomFile.writeBytes(INDENT + INDENT + "setName(\"FIM23\");" +
RETURN);
              randomFile.writeBytes(INDENT + "}" + RETURN);
              randomFile.writeBytes(RETURN);
              randomFile.writeBytes(INDENT + "void buildWord0() {" + RETURN);
              randomFile.writeBytes(INDENT + INDENT + "int Type6Llpvld =1;" +
RETURN);
              randomFile.writeBytes(INDENT + INDENT + "int Type6Altvld =1;" +
RETURN);
              randomFile.writeBytes(INDENT + INDENT + "int Type6Hvvld =1;" +
RETURN);
              randomFile.writeBytes(INDENT + INDENT + "int Type6Hvevld =1;" +
RETURN);
              randomFile.writeBytes(INDENT + INDENT + "int Type6Vvevld =1;" +
RETURN);
               randomFile.writeBytes(INDENT + INDENT + "int Type6Hpevld =1;" +
RETURN);
               randomFile.writeBytes(INDENT + INDENT + "int Type6Altvld1 =1;" +
RETURN);
              randomFile.writeBytes(INDENT + INDENT + "int Type6vpevld  =1;" +
RETURN);
```

```
                randomFile.writeBytes(INDENT + INDENT + "int Type6vvvld =1;" +
RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "int Type6Ttvld =1;" +
RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(Type6Llpvld << 14);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord | Type6Altvld<< 13);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord | Type6Hvvld<< 12);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord | Type6vvvld<< 11);" + RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord | Type6Hvevld  << 10);" + RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord | Type6Vvevld  << 9);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord | Type6Hpevld <<6);" + RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord | Type6vpevld <<5);" + RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord | Type6Ttvld << 3);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord0(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord1() {" + RETURN);
```

```
                randomFile.writeBytes(INDENT + INDENT + "int type6LatMSW =20;" +
RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(type6LatMSW =20);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord1(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord2() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "int type6LatLSW =20;" +
RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(type6LatLSW);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord2(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord3() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord3(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes("}" + RETURN);
               } // end of if FIM23 statement

             //Begining of FIM24
```

```
            if (fims[i] =="FIM24") {
              randomFile.writeBytes("class Fim24 extends Fim {" + RETURN);
              randomFile.writeBytes(INDENT + "short tempWord;" + RETURN);
              randomFile.writeBytes(RETURN);
              randomFile.writeBytes(INDENT + "public Fim24() {" + RETURN);
              randomFile.writeBytes(INDENT + INDENT + "setName(\"FIM24\");" +
RETURN);
              randomFile.writeBytes(INDENT + "}" + RETURN);
              randomFile.writeBytes(RETURN);
              randomFile.writeBytes(INDENT + "void buildWord0() {" + RETURN);
              randomFile.writeBytes(INDENT + INDENT + "int type7TimeTagMSW =
20;" + RETURN);
              randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(type7TimeTagMSW = 20);" + RETURN);
              randomFile.writeBytes(INDENT + INDENT + "setWord0(tempWord);" +
RETURN);
              randomFile.writeBytes(INDENT + "}" + RETURN);
              randomFile.writeBytes(RETURN);
              randomFile.writeBytes(INDENT + "void buildWord1() {" + RETURN);
              randomFile.writeBytes(INDENT + INDENT + "int type7TimeTagLSW =
20;" + RETURN);
              randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(type7TimeTagLSW);" + RETURN);
              randomFile.writeBytes(INDENT + INDENT + "setWord1(tempWord);" +
RETURN);
              randomFile.writeBytes(INDENT + "}" + RETURN);
```

```
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord2() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord2(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord3() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord3(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes("}" + RETURN);
                //
              } // end of if FIM24 statement
             //Begining of FIM25
             if (fims[i] =="FIM25") {
                randomFile.writeBytes("class Fim25 extends Fim {" + RETURN);
                randomFile.writeBytes(INDENT + "short tempWord;" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "public Fim25() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setName(\"FIM25\");" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord0() {" + RETURN);
```

```
            randomFile.writeBytes(INDENT + INDENT + "int type8Reserved1 = 0;" +
RETURN);
            randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(type8Reserved1);" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "setWord0(tempWord);" +
RETURN);
            randomFile.writeBytes(INDENT + "}" + RETURN);
            randomFile.writeBytes(RETURN);
            randomFile.writeBytes(INDENT + "void buildWord1() {" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "int type8Reserved2 =0;" +
RETURN);
            randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(type8Reserved2);" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "setWord1(tempWord);" +
RETURN);
            randomFile.writeBytes(INDENT + "}" + RETURN);
            randomFile.writeBytes(RETURN);

            randomFile.writeBytes(INDENT + "void buildWord2() {" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "int type8TimeTagMSW =
20;" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(type8TimeTagMSW);" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "setWord2(tempWord);" +
RETURN);
            randomFile.writeBytes(INDENT + "}" + RETURN);
```

```
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord3() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "int type8TimeTagLSW =
20;" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(type8TimeTagLSW);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord3(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes("}" + RETURN);
                 //
               } // end of if FIM25 statement
               //Begining of FIM26
               if (fims[i] =="FIM26") {
                randomFile.writeBytes("class Fim26 extends Fim {" + RETURN);
                randomFile.writeBytes(INDENT + "short tempWord;" + RETURN);
                randomFile.writeBytes(RETURN);

                randomFile.writeBytes(INDENT + "public Fim26() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setName(\"FIM26\");" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord0() {" + RETURN);
```

```
            randomFile.writeBytes(INDENT + INDENT + "int type9Reserved1 = 0;" +
RETURN);
            randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(type9Reserved1);" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "setWord0(tempWord);" +
RETURN);
            randomFile.writeBytes(INDENT + "}" + RETURN);
            randomFile.writeBytes(RETURN);
            randomFile.writeBytes(INDENT + "void buildWord1() {" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "int type8Alv = 0;" +
RETURN);
            randomFile.writeBytes(INDENT + INDENT + "int type8Pv = 0;" +
RETURN);
             randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(type8Alv  <<1);" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord | type8Pv  <<8);" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "setWord1(tempWord);" +
RETURN);
            randomFile.writeBytes(INDENT + "}" + RETURN);
            randomFile.writeBytes(RETURN);
            randomFile.writeBytes(INDENT + "void buildWord2() {" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "setWord2(tempWord);" +
RETURN);
            randomFile.writeBytes(INDENT + "}" + RETURN);
            randomFile.writeBytes(RETURN);
```

```
                randomFile.writeBytes(INDENT + "void buildWord3() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord3(tempWord);" +
RETURN);
                 randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes("}" + RETURN);
               } // end of if FIM26 statement
               //Begining of FIM27
               if (fims[i] =="FIM27") {
                randomFile.writeBytes("class Fim27 extends Fim {" + RETURN);
                randomFile.writeBytes(INDENT + "short tempWord;" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "public Fim27() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setName(\"FIM27\");" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord0() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "int SegmentInit1Change1 =
1;" + RETURN);

                randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(SegmentInit1Change1);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord0(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
```

```
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord1() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "int SegmentInit2Change1 =
10;" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(SegmentInit2Change1);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord1(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord2() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord2(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord3() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord3(tempWord);" +
RETURN);
                  randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes("}" + RETURN);
               } // end of if FIM26 statement
               //Begining of FIM28
               if (fims[i] =="FIM28") {
                randomFile.writeBytes("class Fim28 extends Fim {" + RETURN);
                randomFile.writeBytes(INDENT + "short tempWord;" + RETURN);
```

```
            randomFile.writeBytes(RETURN);
            randomFile.writeBytes(INDENT + "public Fim28() {" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "setName(\"FIM28\");" +
RETURN);
            randomFile.writeBytes(INDENT + "}" + RETURN);
            randomFile.writeBytes(RETURN);
            randomFile.writeBytes(INDENT + "void buildWord0() {" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "int SegmentInit1Change2 =
1;" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(SegmentInit1Change2);" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "setWord0(tempWord);" +
RETURN);
            randomFile.writeBytes(INDENT + "}" + RETURN);
            randomFile.writeBytes(RETURN);
            randomFile.writeBytes(INDENT + "void buildWord1() {" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "int SegmentInit2Change2 =
10;" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(SegmentInit2Change2);" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "setWord1(tempWord);" +
RETURN);
            randomFile.writeBytes(INDENT + "}" + RETURN);
            randomFile.writeBytes(RETURN);
            randomFile.writeBytes(INDENT + "void buildWord2() {" + RETURN);
```

```
                randomFile.writeBytes(INDENT + INDENT + "setWord2(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord3() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord3(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes("}" + RETURN);
               } // end of if FIM28 statement
               //Begining of FIM 29
               if (fims[i] =="FIM29") {
                randomFile.writeBytes("class Fim29 extends Fim {" + RETURN);
                randomFile.writeBytes(INDENT + "short tempWord;" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "public Fim29() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setName(\"FIM29\");" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord0() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "int type10RDT =1;" +
RETURN);
                randomFile.writeBytes(INDENT + INDENT + "int type10IFDP =1;" +
RETURN);
```

```
randomFile.writeBytes(INDENT + INDENT + "int type10AlignmentStatus
=1;" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "int
type10InAlignmentMode =1;" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "int type10InModeOperation
=1;" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "int type10BAV  = 0;" +
RETURN);
randomFile.writeBytes(INDENT + INDENT + "int type10WAV  = 0;" +
RETURN);
randomFile.writeBytes(INDENT + INDENT + "int type10VVV  = 0;" +
RETURN);
randomFile.writeBytes(INDENT + INDENT + "int type10VXYV = 0;" +
RETURN);
randomFile.writeBytes(INDENT + INDENT + "int type10PPV  = 1;" +
RETURN);
randomFile.writeBytes(INDENT + INDENT + "int type10TV   = 1;" +
RETURN);
randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(type10RDT << 14);" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord | type10IFDP << 13);" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord | type10AlignmentStatus << 10);" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord | type10InAlignmentMode << 8);" + RETURN);
```

```
                randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord | type10InModeOperation  << 5);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord | type10BAV  << 4);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord | type10WAV << 3);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord | type10VVV  << 2);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord | type10VXYV  << 1);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord | type10PPV  << 1);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord | type10TV);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord0(tempWord);" + RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord1() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "int type10TimeTagMSW = 10;" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(type10TimeTagMSW);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord1(tempWord);" + RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
```

```
                randomFile.writeBytes(INDENT + "void buildWord2() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "int type10TimeTagLSW =
10;" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(type10TimeTagLSW);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord2(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord3() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord3(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                 randomFile.writeBytes(RETURN);
                 randomFile.writeBytes("}" + RETURN);
                 //
               } // end of if FIM29 statement
               //Begining of FIM30
               if (fims[i] =="FIM30") {
                 randomFile.writeBytes("class Fim30 extends Fim {" + RETURN);
                 randomFile.writeBytes(INDENT + "short tempWord;" + RETURN);
                 randomFile.writeBytes(RETURN);
                 randomFile.writeBytes(INDENT + "public Fim30() {" + RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "setName(\"FIM30\");" +
RETURN);
                 randomFile.writeBytes(INDENT + "}" + RETURN);
```

```
            randomFile.writeBytes(RETURN);
            randomFile.writeBytes(INDENT + "void buildWord0() {" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "int type11RDT
=1;" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "int type11SAHRSFDP
=0;" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "int type11SAHRSSync
=1;" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "int type11SAHRSAlli
=1;" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "int
type11SAHRSModeOperation =1;" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "int type11SAHRSNav
=1;" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "int type11BIV
=0;" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "int type11SAHRSTHV
=1;" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "int type11ADCTAOAV
=0;" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "int type11ADCTAV
=1;" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "int type11ADCPAV
=0;" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "int type11TV
=1;" + RETURN);
```

```
                randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(type11RDT  << 14);" + RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord | type11SAHRSFDP << 11);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord | type11SAHRSSync << 10);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord | type11SAHRSAlli << 9);" + RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord | type11SAHRSModeOperation  << 6);" + RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord | type11SAHRSNav << 5);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord | type11BIV  << 4);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord | type11SAHRSTHV   << 3);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord | type11ADCTAOAV  << 2);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord | type11ADCTAV << 1);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord | type11TV );" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord0(tempWord);" + RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord1() {" + RETURN);
```

```
                randomFile.writeBytes(INDENT + INDENT + "int type11TimeTagMSW =
10;" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(type11TimeTagMSW);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord1(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord2() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "int type11TimeTagLSW =
10;" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(type11TimeTagLSW);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord2(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord3() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord3(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes("}" + RETURN);
              } // end of if FIM30 statement
             // Begining of FIM31
             if (fims[i] =="FIM31") {
```

```
            randomFile.writeBytes("class Fim31 extends Fim {" + RETURN);
            randomFile.writeBytes(INDENT + "short tempWord;" + RETURN);
            randomFile.writeBytes(RETURN);
            randomFile.writeBytes(INDENT + "public Fim31() {" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "setName(\"FIM31\");" +
RETURN);
            randomFile.writeBytes(INDENT + "}" + RETURN);
            randomFile.writeBytes(RETURN);
            randomFile.writeBytes(INDENT + "void buildWord0() {" + RETURN);
            //
            randomFile.writeBytes(INDENT + INDENT + "int type12AltUncerntainties
=15;" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "int
type12PosUncerntainties =15;" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "int type12FixType = 1;" +
RETURN);
            randomFile.writeBytes(INDENT + INDENT + "int type12ALTV    =0;" +
RETURN);
            randomFile.writeBytes(INDENT + INDENT + "int type12GPSVV   =0;" +
RETURN);
            randomFile.writeBytes(INDENT + INDENT + "int type12FixPPV  =1;" +
RETURN);
            randomFile.writeBytes(INDENT + INDENT + "int type12TV      =1;" +
RETURN);
            randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(type12AltUncerntainties << 9);" + RETURN);
```

```
                randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord | type12PosUncerntainties <<4);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord | type12FixType  <<4);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord | type12ALTV    <<4);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord | type12GPSVV << 2);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord | type12FixPPV<< 1);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord | type12TV);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord0(tempWord);" + RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord1() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "int type12TimeTagMSW = 10;" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(type12TimeTagMSW);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord1(tempWord);" + RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord2() {" + RETURN);
```

```
                randomFile.writeBytes(INDENT + INDENT + "setWord2(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord3() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord3(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes("}" + RETURN);

              } // end of if FIM31 statement
              //Begining of FIM32
              if (fims[i] =="FIM32") {
                randomFile.writeBytes("class Fim32 extends Fim {" + RETURN);
                randomFile.writeBytes(INDENT + "short tempWord;" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "public Fim32() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setName(\"FIM32\");" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord0() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "int type13RDT =0;" +
RETURN);
```

```
                randomFile.writeBytes(INDENT + INDENT + "int type13IFDP =1;" +
RETURN);
                randomFile.writeBytes(INDENT + INDENT + "int type13InAlignment =1;"
+ RETURN);
                randomFile.writeBytes(INDENT + INDENT + "int type13InRHA       =1;"
+ RETURN);
                randomFile.writeBytes(INDENT + INDENT + "int type13inMHA       =0;"
+ RETURN);
                randomFile.writeBytes(INDENT + INDENT + "int type13inRA        =1;" +
RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "int type13inNA        =1;"
+ RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "int type13NMS         =1;"
+ RETURN);
                randomFile.writeBytes(INDENT + INDENT + "int type13PAV         =0;" +
RETURN);
                randomFile.writeBytes(INDENT + INDENT + "int type13VVV         =1;" +
RETURN);
                randomFile.writeBytes(INDENT + INDENT + "int type13VNEV        =1;"
+ RETURN);
                randomFile.writeBytes(INDENT + INDENT + "int type13LOPV        =0;"
+ RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "int type13LAPV        =1;"
+ RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "int type13TV          =0;" +
RETURN);
```

```
randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(type13RDT << 14);" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord | type13IFDP <<13);" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord | type13InAlignment <<10);" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord | type13InRHA <<9);" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord | type13inMHA <<8);" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord | type13inRA <<7);" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord | type13inNA  <<6);" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord | type13NMS <<5);" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord | type13PAV <<4);" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord | type13VVV <<3);" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord | type13VNEV <<2);" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord | type13LOPV <<1);" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord | type13LAPV <<1);" + RETURN);
```

```
            randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord | type13TV);" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "setWord0(tempWord);" + RETURN);
            randomFile.writeBytes(INDENT + "}" + RETURN);
            randomFile.writeBytes(RETURN);
            randomFile.writeBytes(INDENT + "void buildWord1() {" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "int type13TimeTagMSW = 10;" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(type13TimeTagMSW);" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "setWord1(tempWord);" + RETURN);
            randomFile.writeBytes(INDENT + "}" + RETURN);
            randomFile.writeBytes(RETURN);
            randomFile.writeBytes(INDENT + "void buildWord2() {" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "int type13TimeTagLSW = 10;" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(type13TimeTagLSW);" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "setWord2(tempWord);" + RETURN);
            randomFile.writeBytes(INDENT + "}" + RETURN);
            randomFile.writeBytes(RETURN);
            randomFile.writeBytes(INDENT + "void buildWord3() {" + RETURN);
```

```
                randomFile.writeBytes(INDENT + INDENT + "setWord3(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes("}" + RETURN);
          //
               } // end of if FIM32 statement
               //Beginig of FIM33
               if (fims[i] =="FIM33") {
                randomFile.writeBytes("class Fim33 extends Fim {" + RETURN);
                randomFile.writeBytes(INDENT + "short tempWord;" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "public Fim33() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setName(\"FIM33\");" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord0() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "int type14RDT          =0;"
+ RETURN);
                randomFile.writeBytes(INDENT + INDENT + "int type14SAHRBIV
=1;" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "int type14SAHRTHV
=1;" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "int type14ADCTAOAV
=0;" + RETURN);
```

```
randomFile.writeBytes(INDENT + INDENT + "int type14ADCTAV
=1;" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "int type14ADCPAV
=0;" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "int type14DOPVS
=1;" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "int type14NAVMode
=1;" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "int type14TV =0;" +
RETURN);
randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(type14RDT << 14);" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord | type14SAHRBIV <<6);" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord | type14SAHRTHV <<5);" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord | type14ADCTAOAV << 4);" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord | type14ADCTAV << 3);" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord | type14ADCPAV << 2);" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord | type14DOPVS << 1);" + RETURN);
randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord | type14NAVMode <<1);" + RETURN);
```

```
            randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(tempWord | type14TV);" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "setWord0(tempWord);" + RETURN);
            randomFile.writeBytes(INDENT + "}" + RETURN);
            randomFile.writeBytes(RETURN);
            randomFile.writeBytes(INDENT + "void buildWord1() {" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "int type14TimeTagMSW = 10;" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(type14TimeTagMSW);" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "setWord1(tempWord);" + RETURN);
            randomFile.writeBytes(INDENT + "}" + RETURN);
            randomFile.writeBytes(RETURN);
            randomFile.writeBytes(INDENT + "void buildWord2() {" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "int type14TimeTagLSW = 10;" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "tempWord = (short)(type14TimeTagLSW);" + RETURN);
            randomFile.writeBytes(INDENT + INDENT + "setWord2(tempWord);" + RETURN);
            randomFile.writeBytes(INDENT + "}" + RETURN);
            randomFile.writeBytes(RETURN);
            randomFile.writeBytes(INDENT + "void buildWord3() {" + RETURN);
```

```
                randomFile.writeBytes(INDENT + INDENT + "setWord3(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes("}" + RETURN);

              } // end of if FIM33 statement
              //Begining of FIM34
              if (fims[i] =="FIM34") {
                 randomFile.writeBytes("class Fim34 extends Fim {" + RETURN);
                 randomFile.writeBytes(INDENT + "short tempWord;" + RETURN);
                 randomFile.writeBytes(RETURN);
                 randomFile.writeBytes(INDENT + "public Fim34() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setName(\"FIM34\");" +
RETURN);
                 randomFile.writeBytes(INDENT + "}" + RETURN);
                 randomFile.writeBytes(RETURN);
                 randomFile.writeBytes(INDENT + "void buildWord0() {" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "int RCMessageId =20;" +
RETURN);
                randomFile.writeBytes(INDENT + INDENT + "int RCMessageIdCorrelator
= 5;" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(RCMessageId << 10);" + RETURN);
                  randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord | RCMessageIdCorrelator);" + RETURN);
```

```
                randomFile.writeBytes(INDENT + INDENT + "setWord0(tempWord);" +
RETURN);
                 randomFile.writeBytes(INDENT + "}" + RETURN);
                 randomFile.writeBytes(RETURN);
                 randomFile.writeBytes(INDENT + "void buildWord1() {" + RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "int RCCode =3;" +
RETURN);
                randomFile.writeBytes(INDENT + INDENT + "int RCPriority =5;" +
RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(RCPriority << 11);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "tempWord =
(short)(tempWord | RCCode);" + RETURN);
                randomFile.writeBytes(INDENT + INDENT + "setWord1(tempWord);" +
RETURN);
                randomFile.writeBytes(INDENT + "}" + RETURN);
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes(INDENT + "void buildWord2() {" + RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "setWord2(tempWord);" +
RETURN);
                 randomFile.writeBytes(INDENT + "}" + RETURN);
                 randomFile.writeBytes(RETURN);
                 randomFile.writeBytes(INDENT + "void buildWord3() {" + RETURN);
                 randomFile.writeBytes(INDENT + INDENT + "setWord3(tempWord);" +
RETURN);
                 randomFile.writeBytes(INDENT + "}" + RETURN);
```

```
                randomFile.writeBytes(RETURN);
                randomFile.writeBytes("}" + RETURN);
              } // end of if FIM34 statement
             //End of FIM34
            }
             randomFile.close();  // Done with the file so close it
            }
            catch (Exception e) {
           e.printStackTrace();
            }
          }
        }
```

APPENDIX B- MESSAGE GENERATION FOR PLATFORM 4

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import java.io.*;

class FimTemplate {
  private static void writeData(Fim []fims) {
    File destFile = new File("Test.dat");
    try {
      // Need to delete the old file before writing the new one
      destFile.delete();
      boolean haveLock = destFile.createNewFile();
      RandomAccessFile randomFile = new RandomAccessFile(destFile, "rw");
      for (int i = 0; i < fims.length; i++) {
        if (fims[i].getName().length() > 0) {
          randomFile.writeBytes(fims[i].getName());  // write the name to the file
          randomFile.writeBytes("\r\n");
          randomFile.writeShort(fims[i].getWord0());
          randomFile.writeShort(fims[i].getWord1());
```

```
        randomFile.writeShort(fims[i].getWord2());
        randomFile.writeShort(fims[i].getWord3());
        randomFile.writeBytes("\r\n");
       }
     }
     randomFile.close();  // Done with the file so close it
   }
   catch (Exception e) {
    e.printStackTrace();
   }
  }

  public static void main(String args[]) {
   Fim []fimClasses = new Fim[4];
   fimClasses[0] = new Fim01();
   fimClasses[0].buildWord0();
   fimClasses[0].buildWord1();
   fimClasses[0].buildWord2();

   fimClasses[1] = new Fim02();
   fimClasses[1].buildWord0();
   fimClasses[1].buildWord1();
   fimClasses[1].buildWord2();

   fimClasses[2] = new Fim03();
   fimClasses[2].buildWord0();
   fimClasses[2].buildWord1();
   fimClasses[2].buildWord2();

   fimClasses[3] = new Fim04();
   fimClasses[3].buildWord0();
   fimClasses[3].buildWord1();
   fimClasses[3].buildWord2();

   writeData(fimClasses);
  }
}
abstract class Fim {
  private String name;
  private short word0;
  private short word1;
  private short word2;
```

```
private short word3;

// default constructor
public Fim() {
  // initialize the data
  name = "";
  word0 = 0;
  word1 = 0;
  word2 = 0;
  word3 = 0;
}

public final String getName() {
  return name;
}

public final void setName(String newName) {
  name = newName;
}

public final short getWord0() {
  return word0;
}

public final short getWord1() {
  return word1;
}

public final short getWord2() {
  return word2;
}

public final short getWord3() {
  return word3;
}

public final void setWord0(short newWord) {
  word0 = newWord;
}

public final void setWord1(short newWord) {
  word1 = newWord;
```

```
  }

  public final void setWord2(short newWord) {
    word2 = newWord;
  }

  public final void setWord3(short newWord) {
    word3 = newWord;
  }

  abstract void buildWord0();
  abstract void buildWord1();
  abstract void buildWord2();
  abstract void buildWord3();
}

class Fim01 extends Fim {
  short tempWord;

  public Fim01() {
    setName("FIM01");
  }

  void buildWord0() {
    int carrierMessageId =5;
    int rcmessageId    =2;
    tempWord = (short)(rcmessageId << 10);
    tempWord = (short)(tempWord  |  carrierMessageId);
    setWord0(tempWord);
  }

  void buildWord1() {
    int type         =1;
    int ijms_Select  =1;
    int packingLimit =4;
    int length       =3;
    int priority     =3;
    tempWord = (short)(priority <<11);
    tempWord = (short)(tempWord | (length <<3));
    tempWord = (short)(tempWord | (packingLimit<<5));
    tempWord = (short)(tempWord | (ijms_Select <<2));
    tempWord = (short)(tempWord |  type);
```

```
    setWord1(tempWord);
  }

  void buildWord2() {
    int timeTagMSW   =10;
    int timeTagLSW   =10;
    tempWord = (short)(timeTagMSW<<16);
    tempWord = (short)(tempWord | timeTagLSW <<16);
    setWord2(tempWord);
  }

  void buildWord3() {
  }
}
class Fim02 extends Fim {
  short tempWord;

  public Fim02() {
    setName("FIM02");
  }

  void buildWord0() {
    int requestNumber = 10;
    int requestType = 5;
    int dataWordCount = 7;
    tempWord =  (short)(requestNumber << 8);
    tempWord = (short)(tempWord |  (requestType << 5));
    tempWord = (short)(tempWord | dataWordCount);
    setWord0(tempWord);
  }

  void buildWord1() {
    int coreMemoryAddressMSW = 31;
    tempWord = (short)(coreMemoryAddressMSW);
    setWord1(tempWord);
  }

  void buildWord2() {
    int coreMemoryAddressLSW = 10;
    tempWord = (short)(coreMemoryAddressLSW);
    setWord2(tempWord);
  }
```

```
  void buildWord3() {
    setWord3(tempWord);
  }

}
class Fim10 extends Fim {
  short tempWord;

  public Fim10() {
    setName("FIM10");
  }

  void buildWord0() {
    int drv = 0;
    int wav = 0;
    int av = 0;
    int hv = 0;
    int hvv = 0;
    int vvv = 0;
    int insvcv = 0;
    int instcv = 0;
    int inertial = 0;
    int rdt = 0;
    int rst = 0;
    int gfv = 0;
    int ains = 0;
    tempWord = (short)(drv << 14);
    tempWord = (short)(tempWord |  wav <<13);
    tempWord = (short)(tempWord |  av <<12);
    tempWord = (short)(tempWord |  hv << 11);
    tempWord = (short)(tempWord |  hvv <<10);
    tempWord = (short)(tempWord |  vvv <<9);
    tempWord = (short)(tempWord |  insvcv<<8);
    tempWord = (short)(tempWord |  instcv<<7);
    tempWord = (short)(tempWord |  inertial<<4);
    tempWord = (short)(tempWord |  rdt <<2);
    tempWord = (short)(tempWord |  rst <<1);
    tempWord = (short)(tempWord |  gfv<<1);
    tempWord = (short)(tempWord |  ains);
    setWord0(tempWord);
  }
```

```
  void buildWord1() {
    int gvv = 0;
    int ghv = 0;
    int gfvs= 0;
    int velocitySource = 0;
    int AIP = 0;
    int INSAlignment = 0;
    tempWord = (short)(gvv <<11);
    tempWord = (short)(tempWord | ghv << 10);
    tempWord = (short)(tempWord |  gfvs << 8);
    tempWord = (short)(tempWord |  velocitySource << 4);
    tempWord = (short)(tempWord |  AIP << 2);
    tempWord = (short)(tempWord |  INSAlignment);
    setWord1(tempWord);
  }

  void buildWord2() {
    int deadReckoningLatitude =100;
    tempWord = (short)(deadReckoningLatitude );
    setWord2(tempWord);
  }

  void buildWord3() {
    int deadReckoningLongitude =100;
    tempWord = (short)(deadReckoningLongitude );
    setWord3(tempWord);
  }

}
class Fim15 extends Fim {
  short tempWord;

  public Fim15() {
    setName("FIM15");
  }

  void buildWord0() {
    int trackEntyityType =10;
    int trackIndexNumber =15;
    int trackNumberMSW =2;
    tempWord =  (short)(trackEntyityType<<11);
```

```
    tempWord = (short)(tempWord | trackIndexNumber <<5);
    tempWord = (short)(tempWord | trackNumberMSW);
    setWord0(tempWord);
   }

   void buildWord1() {
    int trackNumberLSW   =15;
    tempWord = (short)(trackNumberLSW);
    setWord1(tempWord);
   }

   void buildWord2() {
    setWord2(tempWord);
   }

   void buildWord3() {
    setWord3(tempWord);
   }

}
```

APPENDIX C- MESSAGE GENERATION FOR PLATFORM 10

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import java.io.*;

class FimTemplate {
  private static void writeData(Fim []fims) {
    File destFile = new File("Test.dat");
    try {
      // Need to delete the old file before writing the new one
      destFile.delete();
      boolean haveLock = destFile.createNewFile();
      RandomAccessFile randomFile = new RandomAccessFile(destFile, "rw");
      for (int i = 0; i < fims.length; i++) {
        if (fims[i].getName().length() > 0) {
          randomFile.writeBytes(fims[i].getName());  // write the name to the file
          randomFile.writeBytes("\r\n");
          randomFile.writeShort(fims[i].getWord0());
```

```
        randomFile.writeShort(fims[i].getWord1());
        randomFile.writeShort(fims[i].getWord2());
        randomFile.writeShort(fims[i].getWord3());
        randomFile.writeBytes("\r\n");
       }
      }
      randomFile.close();  // Done with the file so close it
    }
    catch (Exception e) {
      e.printStackTrace();
    }
  }

  public static void main(String args[]) {
    Fim []fimClasses = new Fim[3];
    fimClasses[0] = new Fim01();
    fimClasses[0].buildWord0();
    fimClasses[0].buildWord1();
    fimClasses[0].buildWord2();

    fimClasses[1] = new Fim02();
    fimClasses[1].buildWord0();
    fimClasses[1].buildWord1();
    fimClasses[1].buildWord2();

    fimClasses[2] = new Fim03();
    fimClasses[2].buildWord0();
    fimClasses[2].buildWord1();
    fimClasses[2].buildWord2();

    writeData(fimClasses);
  }
}
abstract class Fim {
  private String name;
  private short word0;
  private short word1;
  private short word2;
  private short word3;

  // default constructor
  public Fim() {
```

```
  // initialize the data
  name = "";
  word0 = 0;
  word1 = 0;
  word2 = 0;
  word3 = 0;
}

public final String getName() {
  return name;
}

public final void setName(String newName) {
  name = newName;
}

public final short getWord0() {
  return word0;
}

public final short getWord1() {
  return word1;
}

public final short getWord2() {
  return word2;
}

public final short getWord3() {
  return word3;
}

public final void setWord0(short newWord) {
  word0 = newWord;
}

public final void setWord1(short newWord) {
  word1 = newWord;
}

public final void setWord2(short newWord) {
  word2 = newWord;
```

```
  }

  public final void setWord3(short newWord) {
    word3 = newWord;
  }

  abstract void buildWord0();
  abstract void buildWord1();
  abstract void buildWord2();
  abstract void buildWord3();
}

class Fim01 extends Fim {
  short tempWord;

  public Fim01() {
    setName("FIM01");
  }

  void buildWord0() {
    int carrierMessageId =5;
    int rcmessageId    =2;
    tempWord = (short)(rcmessageId << 10);
    tempWord = (short)(tempWord  |  carrierMessageId);
    setWord0(tempWord);
  }

  void buildWord1() {
    int type         =1;
    int ijms_Select  =1;
    int packingLimit =4;
    int length       =3;
    int priority     =3;
    tempWord = (short)(priority <<11);
    tempWord = (short)(tempWord | (length <<3));
    tempWord = (short)(tempWord | (packingLimit<<5));
    tempWord = (short)(tempWord | (ijms_Select <<2));
    tempWord = (short)(tempWord |  type);
    setWord1(tempWord);
  }

  void buildWord2() {
```

```
    int timeTagMSW   =10;
    int timeTagLSW   =10;
    tempWord = (short)(timeTagMSW<<16);
    tempWord = (short)(tempWord | timeTagLSW <<16);
    setWord2(tempWord);
   }

   void buildWord3() {
   }
}
class Fim03 extends Fim {
  short tempWord;

  public Fim03() {
    setName("FIM03");
  }

  void buildWord0() {
    int setIndexNumber = 1;
    tempWord =  (short)(setIndexNumber << 4);
    setWord0(tempWord);
  }

  void buildWord1() {
    int transferSegmentCount = 30;
    tempWord = (short)(transferSegmentCount);
    setWord1(tempWord);
  }

  void buildWord2() {
    setWord3(tempWord);
  }


  void buildWord3() {
    setWord3(tempWord);
  }

}
class Fim15 extends Fim {
  short tempWord;
```

```
 public Fim15() {
  setName("FIM15");
 }

 void buildWord0() {
  int trackEntyityType =10;
  int trackIndexNumber =15;
  int trackNumberMSW =2;
  tempWord =  (short)(trackEntyityType<<11);
  tempWord = (short)(tempWord | trackIndexNumber <<5);
  tempWord = (short)(tempWord | trackNumberMSW);
  setWord0(tempWord);
 }

 void buildWord1() {
  int trackNumberLSW   =15;
  tempWord = (short)(trackNumberLSW);
  setWord1(tempWord);
 }

 void buildWord2() {
  setWord2(tempWord);
 }

 void buildWord3() {
  setWord3(tempWord);
 }

}
```

APPENDIX D-MIDS INTERFACE CONTROL PANEL USER'S MANUAL

Introduction

The User's manual describes the manner in which the MIDS Interface Control Panel is initialized and the manner in which the platform and FIMs are selected and how the code is generated for the platform and MIDS.

Environment

Hardware. The MIDS Interface Control Panel requires a Pentium III or higher microprocessor.

Software. The MIDS interface Control Panel runs on Windows 98 or Windows NT operating systems. You must have a Java compiler installed on your PC. The MIDS Interface Control Panel is compiled using Java 2 SDK version 1.3.1 compiler.

Required steps to compile and run the MIDS Interface Control Panel software are as follows:

Open a DOS window and proceed as follows:

Type “javac Mids_interface_7.java” and then press the return key on the keyboard.

Type “java Mids_interface_7” and then press return to run the Java application, the MIDS Interface Panel will be displayed on the screen.

MIDS Interface Control Panel Button's Description

The MIDS Interface Control Panel has three buttons. The buttons allow the user create to create the desired platform. These buttons are “Select Platform Types”, “Select Messages” and “Code Generator”.

Select Platform Types button:

Put the cursor on the combobox next to the "Select Platform Types" button and click the mouse to choose the platform type from the list (the platform type range is 1 through 31, the initial platform type value is set 6, only one platform can be selected at the time).

Select Messages button:

Put the cursor on the combobox next to the "Select Messages" button and choose the desired a FIM message(s) from the list. FIM01 through FIM34 are available, the initial

FIM value is set to FIM05. Multiple FIMs can be selected by pressing the Ctrl key on the keyboard and clicking the mouse.

Code Generator button:

By pressing the Code Generator button, the code for the platform type and FIMs is generated. A file called FimTemplate.java is generated and can be compiled.

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, VA

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, CA

3. Dr Luqi
   Naval Postgraduate School
   Monterey, California

4. Dr Valdis Berzins
   Naval Postgraduate School
   Monterey, California

5. Dr Nabendu Chaki
   Naval Postgraduate School
   Monterey, California

6. Mr. Milton Mata
   SPAWAR
   San Diego, California

7. SPAWAR, PMW/PMA-159
   San Diego, California